Task-oriented specialization techniques for entity retrieval

Von der Fakultät Informatik, Elektrotechnik und Informationstechnik der Universität Stuttgart zur Erlangung der Würde eines Doktors der Philosophie (Dr. phil.) genehmigte Abhandlung.

Vorgelegt von

Andrea Ulrike Glaser

aus Stuttgart

Hauptberichter	Prof. Dr. Jonas Kuhn
Mitberichter	Prof. Dr. Hinrich Schütze

Tag der mündlichen Prüfung: 24. Juni 2020

Institut für Maschinelle Sprachverarbeitung
der Universität Stuttgart

2020

Erklärung (Statement of Authorship)

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbständig verfasst habe und dabei keine andere als die angegebene Literatur verwendet habe. Alle Zitate und sinngemäßen Entlehnungen sind als solche unter genauer Angabe der Quelle gekennzeichnet.

I hereby declare that this text is the result of my own work and that I have not used sources without declaration in the text. Any thoughts from others or literal quotations are clearly marked.

Ort, Datum

Andrea Glaser

Contents

List of Figures

List of Tables

List of Abbreviations

ARFF	Attribute-Relation File Format
BL	Baseline
BOW	Bag-of-Word
CDCR	Cross-Document Coreference Resolution
CNN	Convolutional Neural Network
CR	Coreference Resolution
D2W	Disambiguation to Wikipedia
EL	Entity Linking
HIT	Human Intelligence Task
HTML	Hyper Text Markup Language
JWPL	Java Wikipdia Library
KB	Knowledge Base
LDA	Latent Dirichlet Allocation
LSTM	Long Short-Term Memory
ML	Machine Learning
MTurk	Amazon Mechanical Turk
NB	Naive Bayes
NE	Named Entity
NED	Named Entity Disambiguation
NEL	Named Entity Linking
NEN	Named Entity Normalization
NER	Named Entity Recognition
NP	Noun Phrase
NLP	Natural Language Processing
POS	Part of Speech
RNN	Recurrent Neural Network
SVM	Support Vector Machines
WSD	Word Sense Disambiguation

Acknowledgments

First and foremost, I would like to thank my adviser Jonas Kuhn for adopting me and my project and integrating me into his research group. I am very grateful for all the interesting discussions that lead to new ideas for this thesis, as well as all the advice and support he gave me during my PhD journey.

I am very thankful to my second reviewer Hinrich Schütze, not only for agreeing to be on my committee but also for getting me interested in pursuing a PhD in the first place. He gave me the opportunity to join his research group in the beginning and I greatly value everything I learned from him.

Grzegorz Dogil's lectures motivated me to join the IMS as a student and I will always be very grateful to him for this.

The IMS was a great place to study and to do research and I would like to thank all my colleagues for making it such a special place. I enjoyed all the discussions we had in the Sentiment Analysis group with Wiltrud Kessler, Christian Scheible, Khalid Al-Khatib, and Charles Jochim. Other colleagues at the IMS include André Blessing, Markus Gärtner, Sabrina Stehwien, Anita Ramm, and Kerstin Jung.

Thank you to the office mates I had during the years and with whom not only I discussed research topics but also had a lot of fun: Patrick Ziering, Wiltrud Kessler, Christian Scheible, Charles Jochim, and Laura Oberländer.

A special thanks goes to my Mensa group, including Patrick Ziering, Max Kisselew, Ina Rösiger, Stefan Müller, and Yvonne Viesel, who made lunch breaks very enjoyable while talking about absolutely everything.

I would like to thank Edgar Hoch for the technical support and Sybille Laderer, Sabine Dieterle, Barbara Schäfer, and Sabine Mohr for dealing with all the administrative issues.

Last but not least, I am very thankful for the support of my family and friends, especially Ulrike Kaufhold, Reem Al-Atrash, Laura Oberländer, Dominike Thomas, Ankita Oswal, and Olga Pirrung.

The work in this thesis was supported by a Nuance Foundation grant and the DFG (SFB 731, project D7).

Abstract

Finding information on the internet has become very important nowadays, and online encyclopedias or websites specialized in certain topics offer users a great amount of information. Search engines support users when trying to find information. However, the vast amount of information makes it difficult to separate relevant from irrelevant facts for a specific information need. In this thesis we explore two areas of natural language processing in the context of retrieving information about entities: named entity disambiguation and sentiment analysis. The goal of this thesis is to use methods from these areas to develop task-oriented specialization techniques for entity retrieval.

Named entity disambiguation is concerned with linking referring expressions (e.g., proper names) in text to their corresponding real world or fictional entity. Identifying the correct entity is an important factor in finding information on the internet as many proper names are ambiguous and need to be disambiguated to find relevant information. To that end, we introduce the notion of *r-context*, a new type of structurally informed context. This r-context consists of sentences that are relevant to the entity only to capture all important context clues and to avoid noise. We then show the usefulness of this r-context by performing a systematic study on a pseudo-ambiguity dataset.

Identifying less known named entities is a challenge in named entity disambiguation because usually there is not much data available from which a machine learning algorithm can learn. We propose an approach that uses an aggregate of textual data about other entities which share certain properties with the target entity, and learn information from it by using topic modelling, which is then used to disambiguate the less known target entity. We use a dataset that is created automatically by exploiting the link structure in Wikipedia, and show that our approach is helpful for disambiguating entities without training material and with little surrounding context.

Retrieving the relevant entities and information can produce many search results. Thus, it is important to effectively present the information to a user. We regard this step beyond the entity retrieval and employ sentiment analysis, which is used to analyze opinions expressed in text, in the context of effectively displaying information about product reviews to a user. We present a system that extracts a *supporting sentence*, a single sentence that captures both the sentiment of the author as well as a supporting

fact. This supporting sentence can be used to provide users with an easy way to assess information in order to make informed choices quickly. We evaluate our approach by using the crowdsourcing service Amazon Mechanical Turk.

Deutsche Zusammenfassung

Das Auffinden von Informationen im Internet ist heutzutage sehr wichtig geworden, und Online-Enzyklopädien und auf bestimmte Themen spezialisierte Webseiten bieten Nutzern eine große Fülle an Informationen. Suchmaschinen unterstützen Benutzer beim Auffinden von Informationen. Allerdings macht die enorme Menge an Informationen es schwieriger, relevante von irrelevanten Fakten für ein spezifisches Infomationsbedürfnis zu unterscheiden. In dieser Dissertation untersuchen wir zwei Bereiche der Maschinellen Sprachverarbeitung im Kontext der Auffindung von Informationen zu Entitäten: Named Entity Disambiguation und Sentimentanalyse. Das Ziel ist es, mit Methoden aus diesen zwei Bereichen anwendungsorientierte Spezialtechniken für das Auffinden von Entitäten zu entwickeln.

Named Entity Disambiguation beschäftigt sich damit, referierende Ausdrücke (z.B. Eigennamen) im Text auf ihre entsprechende Entität in der realen oder fiktiven Welt zu verlinken. Die korrekte Entität zu bestimmen ist ein entscheidender Faktor, um Informationen im Internet zu finden, da viele Eigennamen mehrdeutig sind und desambiguiert werden müssen, um relevante Informationen zu finden. Hierzu führen wir den Begriff *r-context* ein, eine neue Art von strukturell informiertem Kontext. Dieser r-context besteht nur aus den Sätzen, die für die Entität relevant sind, um alle wichtigen Kontexthinweise zu erfassen und ein Rauschen in den Daten zu vermeiden. Danach demonstrieren wir die Nützlichkeit dieses r-contexts, indem wir eine systematische Studie auf einem Pseudoambiguitäten-Datenset durchführen.

Weniger bekannte Entitäten zu identifizieren ist eine Herausforderung, da oft nicht genug Daten vorhanden sind, von denen ein Machine Learning Algorithmus lernen kann. Wir stellen einen Ansatz vor, der ein Aggregat von textuellen Daten verwendet, die bestimmte Eigenschaften mit der Zielentität teilen, und lernen davon Informationen mit Hilfe von Topic Modelling, welches dann dazu verwendet wird, weniger bekannte Zielentitäten zu desambiguieren. Wir verwenden ein Datenset, welches automatisch mit Hilfe der Linkstruktur in Wikipedia erstellt wird, und zeigen, dass unser Ansatz hilfreich ist, um Entitäten ohne Trainingsmaterial und mit wenig umgebenden Kontext zu desambiguieren.

Das Auffinden von relevanten Entitäten und Informationen kann sehr viele Suchergebnisse produzieren. Deshalb ist es wichtig, die Informationen dem Benutzer effektiv zu präsentieren. Wir betrachten diesen Schritt nach dem Auffinden der Entitäten und verwenden Sentimentanalyse, was zum Analysieren von Meinungen im Text genutzt wird, im Kontext von Produktrezensionen und wie Informationen dazu einem Benutzer effektiv angezeigt werden können. Wir präsentieren ein System, das einen *Supporting Sentence* extrahiert, einen einzigen Satz, der sowohl das Sentiment des Autors als auch einen unterstützenden Fakt beinhaltet. Dieser Satz kann verwendet werden, um Benutzern eine einfache Möglichkeit zu geben, auf Informationen zuzugreifen, um schnell informierte Entscheidungen zu treffen. Wir evaluieren unseren Ansatz durch den Crowdsourcing-Service Amazon Mechanical Turk.

1. Introduction

1.1. Motivation

Ever since the beginning of the digital age, the amount of data stored in the world wide web has been growing rapidly. People now have access to a large amount of information websites, such as online encyclopedias that collect a wide range of knowledge about general or more specific interest topics (e.g., Wikipedia, Encyclopædia Britannica Online, The Literary Encyclopedia), individual websites that cover specific topics (e.g., movies, actors, sports), and news websites. Part of this growth of information can be attributed to websites and services that publish user-generated content, for example, blogs, forums, or platforms with a review feature (e.g., Amazon, IMDb).

A larger amount of data provides more information, but is only helpful if users can find answers that are relevant to their specific information need in the vast amount of irrelevant information. To access these amounts of data more efficiently and to find the information that one is interested in, information retrieval (IR) methods play an important part in search engines. Web search engines (e.g., Google, Bing) employ web crawlers in their backend to regularly go through web pages on the internet and to create a document index that allows fast access to these web pages. On the frontend side, users are provided with an interface where they can enter a search query in the form of keywords or sentences. This query is then processed with natural language processing (NLP) methods, such as stemming and removing stopwords. In the next step, the IR system uses the document index and the keywords from the query to retrieve web pages with information which is relevant to the query.

A typical example of a user's information need is finding information about named entities (e.g., people, locations, organizations). The search query might consist of the named entity only, have additional keywords that can serve as context[1], or might contain keywords only if the named entity is not known. To satisfy the information need, the search engine needs to return web pages that contain information about the specific

[1]Guo et al. (2009) analyzed 1,000 randomly selected web queries from a "commercial web search engine" and found that 71% of the queries contained named entities. Yin and Shah (2010) state that in an internal study of Microsoft, 20-30% of the queries submitted to Bing were named entities only without further context.

entity (and keyword context of the query, if applicable). The relevance of a web page given the user query is determined by the context of the web page among other things. For obtaining information relevant to a specific named entity it is also necessary to identify the correct entity.

In this thesis, we address two NLP research areas in the context of retrieving information about entities. The first one is linking name expressions in text to their corresponding real world or fictional entities. This is an important step to identify the correct entity to increase the relevance of the information of the retrieved documents. The second one is using sentiment analysis to provide more information about an entity, namely whether or not certain people have a positive or negative attitude towards the entity, and what aspects they appreciate or dislike about it. The goal of this thesis is to develop task-oriented specialization techniques for entity retrieval with linguistically informed methods. We investigate three issues which tend to be considered of subordinate concern in standard retrieval approaches, but can improve the user's experience in these specific cases:

Improved accuracy of results. Many search queries are about specific entities, such as persons, locations, or organizations. A general problem that retrieval systems face is the fact that many name expressions are ambiguous and can refer to different entities. For example, the name *"Michael Jackson"* can refer to the famous American singer, but also to many other persons with the same name. Other name expressions are even more ambiguous. For example, the name *"Paris"* can refer to the capital of France but also to cities in other countries, given names or surnames of people, movies, or songs, among other things. These ambiguous name expressions need to be disambiguated to be connected to the correct entity.

Retrieving relevant information is challenging in cases of ambiguous name expressions: if the system cannot correctly identify an entity in a document, the retrieval system will return documents with non-relevant information about other entities. Depending on how the context of the documents is processed, the system might be too lenient (i.e., might return more documents about the wrong entity) or too strict (i.e., will miss documents with relevant information). In an ideal scenario, a system will identify the correct corresponding entity and return only information relevant to it. We approach the problem from a linguistic point of view and propose a method that improves correct identification of entities by making informed context choices, which improves the overall exactness of results returned by a system.

Better findability of rare cases. A special sub-case of entity retrieval and an additional challenge is that in general, retrieval systems work reasonably well for retrieving information about entities that are well-known but often fail when searching for entities which are little-known. In such cases, the system is more likely to return no results or erroneous ones if it cannot identify the unknown entity properly. That is, the system either fails to identify the entity or it confuses it for another more prominent entity. Consequently, the top search results might contain results about non-relevant well-known entities whereas relevant information about the less known entity is further down in the results or not retrieved at all.

Search engines like Google provide users with a list of search results, ranked by certain criteria that determine how relevant websites are to the user's query. Google disclosed in their help pages[2] that they use over 200 criteria to determine relevance. They do not provide a public list of these criteria except for PageRank (Page et al., 1998), an algorithm that determines the relevance of a website based on the number and importance of the incoming links from other websites. Each incoming link is weighted by the importance of the website (where importance is defined by the PageRank value of the website). A higher PageRank value means that the website will be ranked higher in the search result. Well-known entities often appear on more web pages in general and on web pages with a higher PageRank, thus are ranked higher in the search results.

An important general factor for determining relevance is identifying the correct entity of the name expression in the search query and in the documents. The correct identification of little-known entities can be hard for a system if documents about these entities contain only little relevant context. Since in most cases search queries are about well-known entities, developing robust methods to find more obscure information has often been neglected. However, it is problematic for any system to systematically miss important information. We address this issue in this thesis and present an approach that uses known characteristics of entities to identify little-known entities.

Effective presentation of results. Identifying relevant information for a user's query is only one part of a good retrieval system. Very often, there is not only one result for the query but many results. Consequently, the presentation of the retrieved information to the user becomes an important part of the system as it should offer a way to easily and effectively process the information. As mentioned above, search engines often provide lists of ranked results based on different criteria. In addition to ranking the results, search engines often add previews of the web pages in the form of small snippets to give the user a quick way to decide whether a link is interesting enough to explore further. These previews often display the context surrounding the keywords of the search query

[2]`https://web.archive.org/web/20170518231308/https://support.google.com/webmasters/answer/70897`

as it appears in the document.

There are scenarios in which general search engines help find information, but for certain tasks, a more specialized retrieval system would be more efficient. An example is finding information about products through reviews. Such reviews can be written by users or experts and be uploaded to platforms that publish reviews. These can be websites that offer the products for sale or as a service (e.g., Amazon), but reviews can also be published through blogs or social media (e.g., Twitter). Websites with a review system usually provide ways to filter reviews according to different criteria to the users. However, filters are often limited to options like “most recent reviews”, “most helpful reviews”, or “all reviews with a rating of 5”. A user still has to read through the filtered amount of data to find the information they are interested in. This can be a very time-consuming task when there are hundreds of reviews on one website, or when a product is less known and reviews have to be looked up on different websites. We define a method that extracts and displays a relevant snippet of each review so that users can quickly decide whether they want to read more of the review. This system is not limited to reviews on a single website but can work in a review search engine that collects and analyzes reviews from different websites.

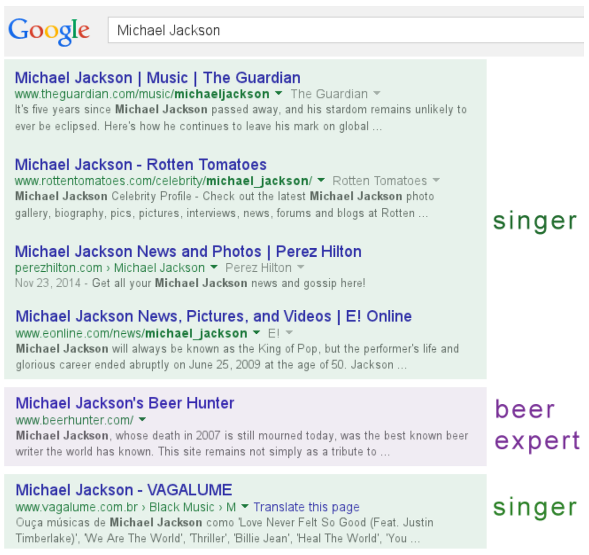
In the next section we describe these three mentioned issues in more detail and propose solutions to address and resolve them.

1.2. Research questions and contributions

The goal of this thesis is to develop task-oriented specialization techniques for entity retrieval with linguistically informed methods. We investigate the three issues mentioned in the previous section and propose methods for resolving them. In this section we pose research questions, each of them corresponding to one of the issues we raise. We describe them in more detail and indicate how we attempt to resolve them in this thesis.

Improved accuracy of results

Retrieval systems often employ complex techniques to obtain the most relevant results and rank them accordingly. This becomes more difficult if a search query is ambiguous as is the case with proper names. Unless someone has a very unique name (e.g., *“Henry VIII of England”*), most proper names are shared by many people. If a search query only contains the proper name, it is not clear for a retrieval system which person the user is looking for. In many cases there is one famous person with the name, for example, *“Michael Jackson, the American singer”*, while other people with the same name are



Michael Jackson

Michael Jackson | Music | The Guardian
www.theguardian.com/music/**michaeljackson** The Guardian
It's five years since **Michael Jackson** passed away, and his stardom remains unlikely to ever be eclipsed. Here's how he continues to leave his mark on global ...

Michael Jackson - Rotten Tomatoes
www.rottentomatoes.com/celebrity/**michael_jackson**/ Rotten Tomatoes
Michael Jackson Celebrity Profile - Check out the latest **Michael Jackson** photo gallery, biography, pics, pictures, interviews, news, forums and blogs at Rotten ...

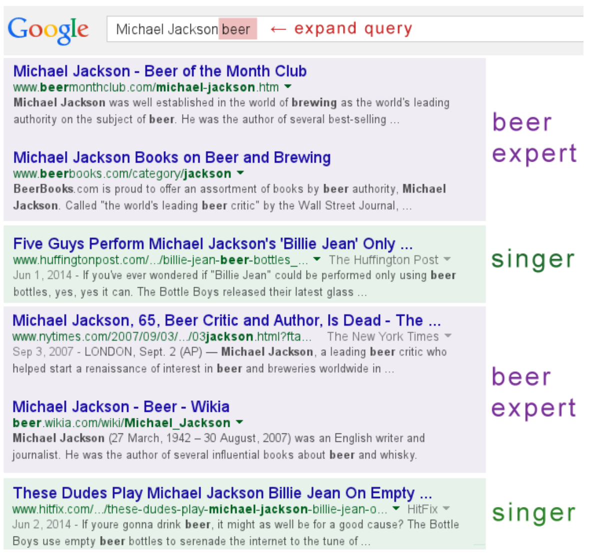
Michael Jackson News and Photos | Perez Hilton
perezhilton.com › Michael Jackson Perez Hilton
Nov 23, 2014 - Get all your **Michael Jackson** news and gossip here!

Michael Jackson News, Pictures, and Videos | E! Online
www.eonline.com/news/**michael_jackson** E!
Michael Jackson will always be known as the King of Pop, but the performer's life and glorious career ended abruptly on June 25, 2009 at the age of 50. Jackson ...

Michael Jackson's Beer Hunter
www.beerhunter.com/
Michael Jackson, whose death in 2007 is still mourned today, was the best known beer writer the world has known. This site remains not simply as a tribute to ...

Michael Jackson - VAGALUME
www.vagalume.com.br › Black Music › M Translate this page
Ouça músicas de **Michael Jackson** como 'Love Never Felt So Good (Feat. Justin Timberlake)', 'We Are The World', 'Thriller', 'Billie Jean', 'Heal The World', 'You ...

(a) Original query *"Michael Jackson"*



Michael Jackson beer

Michael Jackson - Beer of the Month Club
www.**beer**monthclub.com/**michael-jackson**.htm
Michael Jackson was well established in the world of **brewing** as the world's leading authority on the subject of **beer**. He was the author of several best-selling ...

Michael Jackson Books on Beer and Brewing
www.**beer**books.com/category/**jackson**
BeerBooks.com is proud to offer an assortment of books by **beer** authority, **Michael Jackson**. Called "the world's leading **beer** critic" by the Wall Street Journal, ...

Five Guys Perform Michael Jackson's 'Billie Jean' Only ...
www.huffingtonpost.com/.../billie-jean-**beer**-bottles_... The Huffington Post
Jun 1, 2014 - If you've ever wondered if "Billie Jean" could be performed only using **beer** bottles, yes, yes it can. The Bottle Boys released their latest glass ...

Michael Jackson, 65, Beer Critic and Author, Is Dead - The ...
www.nytimes.com/2007/09/03/.../03**jackson**.html?fta... The New York Times
Sep 3, 2007 - LONDON, Sept. 2 (AP) — **Michael Jackson**, a leading **beer** critic who helped start a renaissance of interest in **beer** and breweries worldwide in ...

Michael Jackson - Beer - Wikia
beer.wikia.com/wiki/**Michael_Jackson**
Michael Jackson (27 March, 1942 – 30 August, 2007) was an English writer and journalist. He was the author of several influential books about **beer** and whisky.

These Dudes Play Michael Jackson Billie Jean On Empty ...
www.hitfix.com/.../these-dudes-play-**michael-jackson**-billie-jean-o... HitFix
Jun 2, 2014 - If youre gonna drink **beer**, it might as well be for a good cause? The Bottle Boys use empty **beer** bottles to serenade the internet to the tune of ...

(b) Expanded query *"Michael Jackson beer"*

Figure 1.1.: Google results for the search query *"Michael Jackson"* (green results) and the expanded query *"Michael Jackson beer"* (purple results). Screenshots taken on December 2, 2014. Color highlighting and labeling in the results and the query field are not part of Google's results page and were done manually.

less known. This leads to two problems when searching for a *"Michael Jackson"* that is not well-known: (i) most websites are about the famous person and (ii) websites about famous people usually have a higher PageRank, thus are ranked higher in search results, making a user look through several pages until he or she might find relevant websites.

Figure 1.1 shows results for the search query *"Michael Jackson"* and the expanded search query *"Michael Jackson beer"* on Google. The color highlighting in the results (green for the singer and purple for the beer expert) as well as the query field are not part of Google's results page and were added manually to better show which entities are returned in the results of the respective searches.

Figure 1.1a shows results for the simple search query *"Michael Jackson"*. Most results on the first few pages are about the American singer, and even on later pages he still dominates the search results. This is expected behavior as the query does not contain any further context and web pages about popular entities have a high PageRank which makes them appear higher in the search results. To obtain more precise search results about a different entity, the initial query can be expanded with more specific keywords about the target person. For example, if someone is interested in finding websites about *"Michael Jackson, the beer expert"*, the query can be expanded with words like *"beer"* or *"beer expert"*. This additional context can help find more relevant documents by comparing the keywords in the query with the context of the documents. However, this might still return unwanted results about other people as can be seen in Figure 1.1b. This can happen if the query words appear somewhere in the same document but are not part of the prominent relevant characteristics that describe the person. For example, when expanding the query with *"beer"*, then search results about the American singer can show up because there were five peope who performed one of Michael Jackson's songs on beer bottles.

Context plays an important role when identifying a proper name in a text. Previous work has often used approaches based on a very simple operationalization of the relevant context. For example, using the entire document (e.g., Bollegala et al. (2006); Ikeda et al. (2009); Hoffart et al. (2011)) provides a great amount of context for a comparison with the keywords from the search query. However, since a document often contains information about other entities as well, the information can be noisy. Another common approach is to use a fixed window of words or n-grams around the name (e.g, Pedersen et al. (2005); Bunescu and Pasça (2006); Nguyen and Cao (2008); Ikeda et al. (2009); Han et al. (2011); Li et al. (2013)). These approaches assume that important information is located very closely around the entity. While this is true in many cases, such approaches fail to obtain relevant context if the important information is located farther away in the document. Consider the following sequence of sentences, taken from a news article[3]:

(1.1) S_1: *"Filmmaker J.J. Abrams is adapting a new book about* ***Michael Jackson*** *for television with TV and radio host Tavis Smiley, co-author of the book."*

S_2: *"Smiley's company said Monday that Warner Bros. Television is on board the project with Smiley and Abrams."*

S_3: *"The book, "Before You Judge Me: The Triumph and Tragedy of Michael Jackson's Last Days," will be published Tuesday by Little, Brown and Co."*

S_4: *"Written by Smiley and David Ritz, it's described as a novelistic take on* ***the pop star****'s final months."*

S_5: *"****Jackson****, on the brink of a career comeback, died in June 2009 from an overdose of sedatives."*

S_6: *"Abrams and Smiley will be executive producers on the TV series. A premier date and network that will air it wasn't announced."*

S_7: *"Abrams' credits include "Star Wars" and "Star Trek" movies."*

The first sentence (S_1) mentions a person with the name *"Michael Jackson"*, but there are no distinct cues in the sentence that could identify the entity. The only information

[3]`https://www.nbcchicago.com/entertainment/entertainment-news/TV-Series-on-Michael-Jacksons-Final-Days-in-Development-383713531.html`

available is that there is a book about this Michael Jackson, but this is not a clear identification feature as there are books written about more than one Michael Jackson. The next two sentences elaborate on the planned movie and the people involved in making it. Approaches that take only a fixed window around an entity would not gather useful information in this case. Moreover, since there are many keywords that have to do with television, this Michael Jackson could be misclassified as the television executive or talk radio host. Only much later, in the fifth sentence (S_5), there is another mention of *"Michael Jackson"*, and the sentence contains information that helps identify the entity. Sentence S_4 contains a mention of *"Michael Jackson"* in form of a common noun (*"the pop star"*). This sentence might give helpful information too with the mention of his "final months".

This leads us to our first research question:

Research Question 1 How helpful is it for named entity disambiguation to expand context in a linguistically informed way by adding the context of expressions from the same document that are (automatically) identified as coreferent?

To address this question we conduct an exploratory study which focuses on the following hypotheses:

1. A larger window around mentions of names provides more reliable feature information.
2. Structurally informed choice of the context window helps improve the disambiguation.
3. Identifying coreference chains with names provides relevant context.

In our work we systematically explore the effects of different sizes and types of context. We propose a new type of context that does not take the entire document or a fixed window around the entity but is created by extracting sentences that are relevant to the entity. For this, we use coreference information that we obtain by applying a coreference resolution system on our data. We experiment with different types of coreferent mentions to create different contexts, and compare them to simple context approaches (entire document and fixed window contexts).

We use two kinds of annotations for training and evaluation: (i) referent identification of proper names and (ii) coreference information. One challenge we faced was the lack of a large enough dataset annotated with both types of information. Furthermore, annotating a new and adequately large dataset would have been expensive. To solve this problem we created a dataset with pseudo-ambiguous mentions of proper names and automatic annotation of coreference information. This pseudo-ambiguity dataset gives us an analytical advantage: it allows us to experiment with enough data to study different corpus sizes as training material without the need to annotate a large dataset by hand. Pseudo-ambiguity has been introduced for word sense disambiguation (Schütze, 1992; Gale et al., 1992) and has later been used for disambiguating named entities (e.g., Mann and Yarowsky (2003); Pedersen et al. (2005)). For a comparison of pseudo-ambiguity data and real data, we annotated a small dataset with information about ambiguous entities by hand (while the coreference annotation was done automatically for all datasets).

Automatic annotation techniques for coreference information are not perfect since coreference resolution is a task that has yet to be solved (Ng, 2017). However, in a real scenario no coreference gold labels are available and documents need to be labeled automatically. Thus, using automatic annotation of coreference information gives a clearer picture of how our approach works on data in a real scenario.

We show that our approach of using relevant context improves disambiguation results in most cases compared to approaches that use the entire document or a fixed window around the entity. In our error analysis we show why our approach sometimes does not improve the results.

To summarize, our main contributions towards research question 1 are as follows:

- We show that our approach of using structurally informed choice of context generally helps improve the disambiguation of proper names.
- We describe the process for creating a large pseudo-ambiguity dataset that can be used for experiments and show that the results are comparable to those performed on real data.
- We perform a systematic study with different parameters (corpus size, corpus creation method, context size and types) on the pseudo-ambiguity dataset and present detailed results which show the effects of these parameters.

Better findability of rare cases

It is generally easy to find information about well-known entities because there is usually a great amount of information available. This makes it easier for retrieval systems to find existing information that is relevant. However, this is usually not the case for less known entities. Since little information exists about these entities (in databases or the internet in general), it is hard for a retrieval system to find the relevant information.

In practice, people are more interested in well-known entities. Google provides a service called Google Trends[4], which shows statistics about what people are searching for when using the Google search engine. Results can be filtered by region, time range, categories, and search service being used (e.g., web search or picture search).

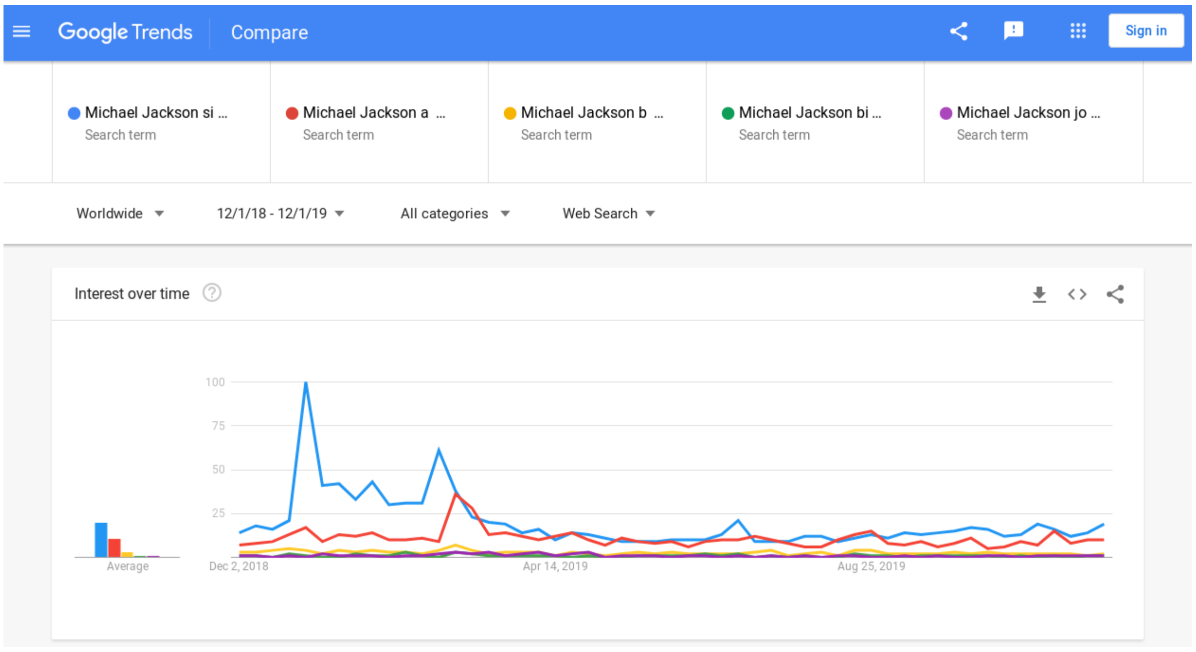


Figure 1.2.: Google trends comparison for five search queries: *Michael Jackson singer* (blue), *Michael Jackson actor* (red), *Michael Jackson beer* (yellow), *Michael Jackson bishop* (green), *Michael Jackson journalist* (purple); using worldwide statistics for the web search over a period of 12 months (December 1, 2018 – December 1, 2019). Numbers represent search interest relative to the highest point on the chart. A value of 100 is the peak popularity for the term. A value of 50 means that the term is half as popular. A score of 0 means there was not enough data for this term. Screenshot taken on December 2, 2019.

Figure 1.2 shows a screenshot of such a comparison for five different search queries with the proper name *"Michael Jackson"*: *Michael Jackson singer* (blue), *Michael Jackson actor* (red), *Michael Jackson beer* (yellow), *Michael Jackson bishop* (green), *Michael Jackson journalist* (purple). Out of these five queries[5], *Michael Jackson singer* is the most searched query (interest average of 20), followed by *Michael Jackson actor* (interest

[4]`https://trends.google.com/`

[5]This comparison does not include the simple search query *Michael Jackson*, which is used much more often: if we include it in the graph, the interest average for it is 25, while it is between 0 and 1 for the other five other search queries. If we assume that people who search for the singer find information immediately with the simple search query, while for other entities the query needs to be expanded, the search intent for the singer is much higher than in the given graph.

average of 11). The other three queries are searched far less with interest averages of 3 (*Michael Jackson beer*), 1 (*Michael Jackson bishop*), and 1 (*Michael Jackson journalist*). This shows that less known entities are much less searched compared to well-known entities in a search engine like Google.

Since most searches performed are for well-known entities, retrieval systems can achieve a high accuracy while neglecting rare cases. However, it is problematic for a system if it cannot correctly identify the target of a less known entity. The result is either missing information or incorrectly mapping the entity to a more well-known one, which results in returning the wrong information. To improve on this fact, we propose a method that aims at identifying the less known entities.

We address this problem with our second research question:

Research Question 2 How can we learn general properties that allow us to identify a little-known entity without having seen the entity in context before?

We investigate this question in the context of proper names and develop a system that can identify the non-prominent name bearers. The idea is that even though we do not have training material for these less known entities, we can create a special type of training material that shares the same characteristic properties as these entities (e.g., profession and nationality). For the experiments, we use a system that incorporates topic models following the latent topic modelling approach based on Blei et al. (2003), which induces a set of so-called topics from a document collection in an unsupervised way.

Our main contributions towards research question 2 are as follows:

- We demonstrate that it is possible to identify little-known entities (for which however certain properties are known) without having explicit training material for them, by using some aggregate of textual data that shares the same properties.
- We show how to create this aggregate of text data for several properties, as well as how to create a silver standard corpus that can be used for evaluation.
- We perform a systematic study with different parameters (corpus type, context size, number of topics used for the topic model) and present detailed results which show the effects of these parameters.
- We show that our approach is indeed very helpful for disambiguating (i) entities for which not much training material is available, and (ii) for entities with little surrounding context, both of wich are useful for many applications.

Effective presentation of results

It is important to not only retrieve the information which is relevant to the user's information need, but also how to present this information. This is especially important if the system returns several results for a user's query and if the list of results contains false positives, i.e., results that do not contain relevant information. Going through many links or reading many documents can be tedious and time-consuming.

The third issue we investigate goes beyond the step of retrieving entities. We assume that the entities have already been extracted and focus on displaying them together with some information. To investigate this issue, we focus on a special type of entity, namely products. Previous work on recognizing entities often focuses on classes such as `person`, `location`, and `organization`. However, more fine grained recognition includes classes such as `products` (e.g., Sarmento et al. (2006); Zhao and Liu (2008); Kravalová and Žabokrtský (2009); Bontcheva et al. (2017)), which covers, for example, books, movies, or technical devices.

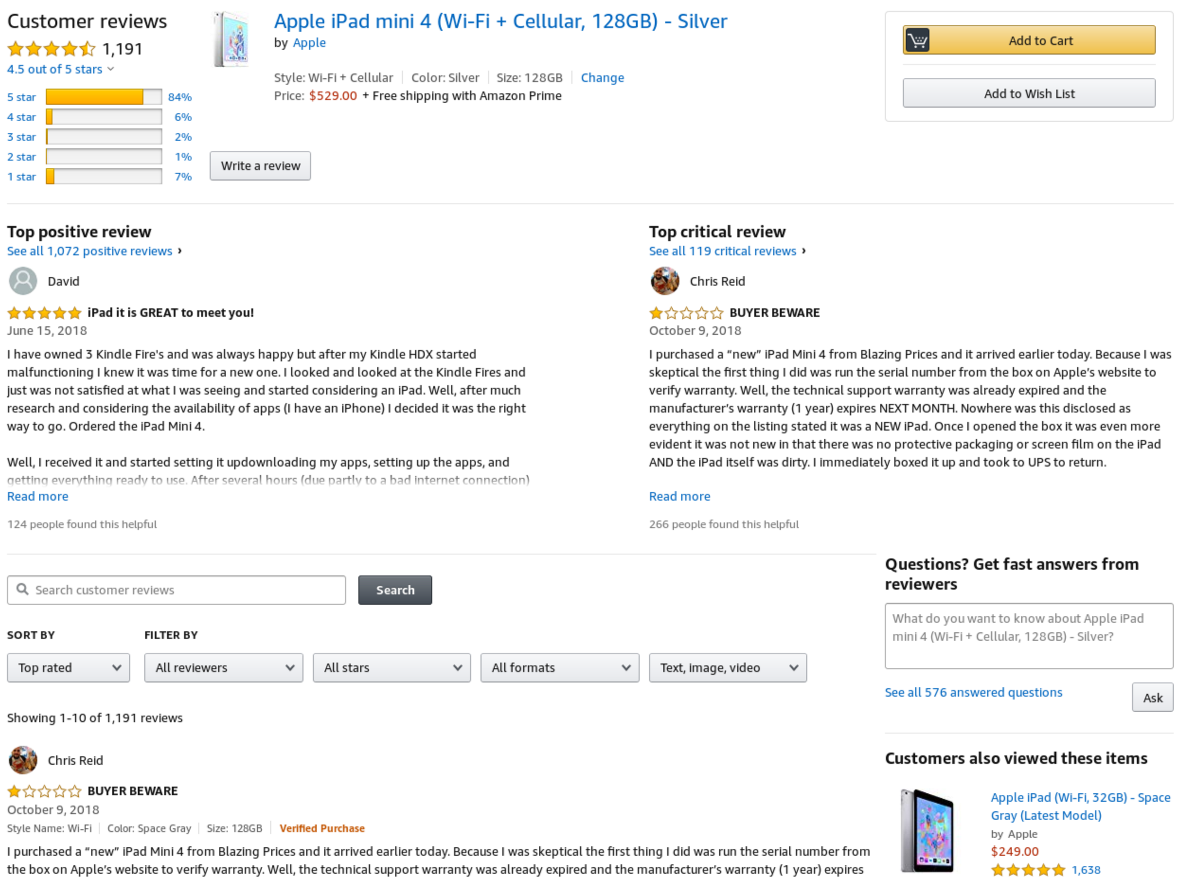
The internet has become a useful tool for deciding whether a product is to a user's liking as many websites provide a platform for users and/or experts to publish their reviews, and explain why or why not they like a certain product (e.g., IMDb for movie reviews or Amazon for reviews about a great variety of products). These websites display a large amount of information about products. However, some products can have several hundreds or several thousand of reviews, which makes finding specific information the product harder.

Figure 1.3 shows an example customer review page on the platform `amazon.com`, with a total of 1,191 reviews. The top of the page features the "top positive review" (the 5-star review which was voted helpful by most people) and the "top critical review" (the 1-star review which was voted helpful by most people). The remaining reviews can be displayed by different sorting and filter methods, which are shown at the bottom of Figure 1.3. However, after applying the filters, users still need to read through the complete review to find information.

Another feature that Amazon provides is called *"Read reviews that mention"*, as shown in Figure 1.4. This feature displays about 5-20 phrases that are mentioned in reviews[6]. While the feature can help users find information in reviews, it is very limited. Expressions such as *"highly recommended"*, *"love my ipad"*, or *"absolute love"* do not specify which features of the product are good or bad. Another downside is that the mentions do not show whether people liked the feature or not.

User-written reviews are also interesting for companies because they provide them

[6]There is no publicly available information about how the feature works exactly.



Customer reviews

1,191

4.5 out of 5 stars

5 star	84%
4 star	6%
3 star	2%
2 star	1%
1 star	7%

Write a review

Apple iPad mini 4 (Wi-Fi + Cellular, 128GB) - Silver

by Apple

Style: Wi-Fi + Cellular | Color: Silver | Size: 128GB | Change

Price: $529.00 + Free shipping with Amazon Prime

Add to Cart

Add to Wish List

Top positive review

See all 1,072 positive reviews ›

David

iPad it is GREAT to meet you!

June 15, 2018

I have owned 3 Kindle Fire's and was always happy but after my Kindle HDX started malfunctioning I knew it was time for a new one. I looked and looked at the Kindle Fires and just was not satisfied at what I was seeing and started considering an iPad. Well, after much research and considering the availability of apps (I have an iPhone) I decided it was the right way to go. Ordered the iPad Mini 4.

Well, I received it and started setting it updownloading my apps, setting up the apps, and getting everything ready to use. After several hours (due partly to a bad internet connection)

Read more

124 people found this helpful

Top critical review

See all 119 critical reviews ›

Chris Reid

BUYER BEWARE

October 9, 2018

I purchased a "new" iPad Mini 4 from Blazing Prices and it arrived earlier today. Because I was skeptical the first thing I did was run the serial number from the box on Apple's website to verify warranty. Well, the technical support warranty was already expired and the manufacturer's warranty (1 year) expires NEXT MONTH. Nowhere was this disclosed as everything on the listing stated it was a NEW iPad. Once I opened the box it was even more evident it was not new in that there was no protective packaging or screen film on the iPad AND the iPad itself was dirty. I immediately boxed it up and took to UPS to return.

Read more

266 people found this helpful

Search customer reviews | Search

SORT BY | FILTER BY

Top rated | All reviewers | All stars | All formats | Text, image, video

Showing 1-10 of 1,191 reviews

Chris Reid

BUYER BEWARE

October 9, 2018

Style Name: Wi-Fi | Color: Space Gray | Size: 128GB | Verified Purchase

I purchased a "new" iPad Mini 4 from Blazing Prices and it arrived earlier today. Because I was skeptical the first thing I did was run the serial number from the box on Apple's website to verify warranty. Well, the technical support warranty was already expired and the manufacturer's warranty (1 year) expires

Questions? Get fast answers from reviewers

What do you want to know about Apple iPad mini 4 (Wi-Fi + Cellular, 128GB) - Silver?

See all 576 answered questions | Ask

Customers also viewed these items

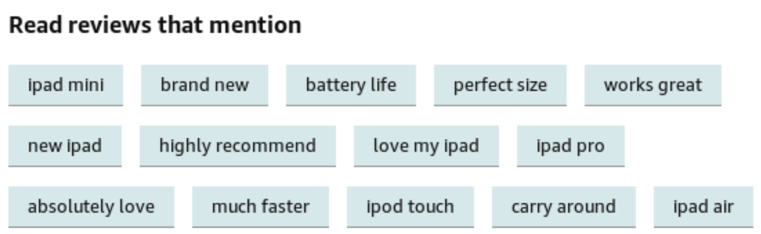
Apple iPad (Wi-Fi, 32GB) - Space Gray (Latest Model)

by Apple

$249.00

1,638

Figure 1.3.: Example review page about a specific product (Apple iPad mini 4) on `amazon.com`, top part is shown. Screenshot taken on March 4, 2019.



Read reviews that mention

ipad mini | brand new | battery life | perfect size | works great

new ipad | highly recommend | love my ipad | ipad pro

absolutely love | much faster | ipod touch | carry around | ipad air

Figure 1.4.: "Read reviews that mention" feature on `amazon.com` for a specific product (Apple iPad mini 4). Screenshot taken on March 4, 2019.

information as to why users of their product like or dislike it, and which specific features they like or dislike. Companies may not only check one website (e.g., Amazon), but obtain reviews about their products from different sources. In such cases the amount of information is even higher and it is even more important to have an effective way of displaying the information.

This leads us to our final research question in this thesis:

Research Question 3 How can opinions about entities be presented in an effective way and so that users can see not only the sentiment (positive/negative) but also a reason the reviewer provides for the sentiment?

We address this research question in the context of user-generated product reviews, published on websites such as Amazon. We present a system that can extract sentences that contain (i) the sentiment of the sentence (which will also be representative for the sentiment of the entire review) and (ii) a reason why the author thinks the product is good or bad. We call such a sentence a *supporting sentence* because it supports the author's assessment about whether the product is good or bad. Supporting sentences contain convincing reasons that provide detailed positive or negative facts about the product or its aspects. The goal is to provide users with an easy way to assess information that contains sentiment as well as reasons for the sentiment so they can make informed choices quickly.

Analyzing reasons and evaluating whether they convincingly support a sentiment is not an easy task. Furthermore, annotating a large dataset requires substantial effort. In our work we define a novel evaluation methodology and explore whether a crowdsourcing service like Amazon Mechanical Turk can be exploited to evaluate this type of problem.

Our main contributions towards research question 3 are as follows:

- We present a system that extracts supporting sentences from reviews—a sentence that includes the sentiment of the overall review as well as a supporting fact for the writer's opinion—which can be used to quickly skim many reviews and find the information a user is looking for.
- We show how a crowdsourcing service like Amazon Mechanial Turk can be used to obtain a large amount of annotations without the need to hire experts and discuss design questions and quality control measures for using such a service.

Bringing it all together

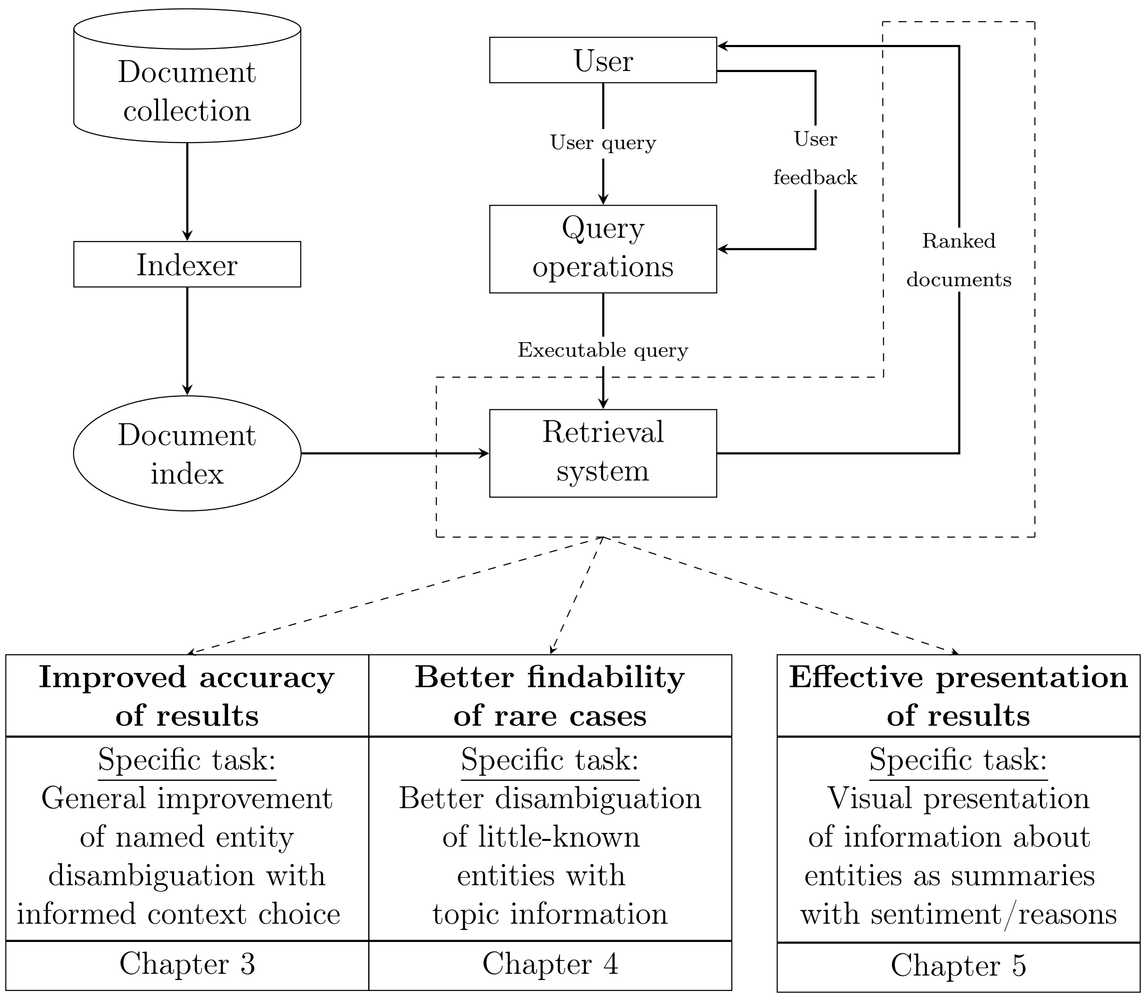

Document collection
Indexer
Document index
User
User query
Query operations
User feedback
Executable query
Retrieval system
Ranked documents
Improved accuracy of results
Specific task: General improvement of named entity disambiguation with informed context choice
Chapter 3
Better findability of rare cases
Specific task: Better disambiguation of little-known entities with topic information
Chapter 4
Effective presentation of results
Specific task: Visual presentation of information about entities as summaries with sentiment/reasons
Chapter 5

Figure 1.5.: Top part: A general information retrieval architecture (Liu, 2011, p.213). Bottom part: The three issues we address in this thesis in the context of the specific tasks.

To summarize, the work in this thesis is done in the context of retrieving information about entities. The goal is to develop task-oriented specialization techniques that help improve the retrieval process and the presentation of retrieved results with linguistically informed methods. We address two NLP research areas in this context: (i) linking name expressions in text to their corresponding real world or fictional entities to obtain better results for entity retrieval, and (ii) using sentiment analysis to provide more information about an entity.

Figure 1.5 shows a general information retrieval architecture (Liu, 2011, p.213). A document index is created from a collection of documents to retrieve relevant documents, depending on the user's query. If the user does not obtain the desired information, they can give feedback (e.g., changing the query). The parts of the architecture we are concerned with in this thesis are the retrieval system part and how the results are presented to users (dashed box in Figure 1.5). We investigate specific tasks in retrieval system scenarios and propose methods for the parts within the dashed box, that help (i) improve accuracy of results in general, (ii) improve findability of rare cases, and (iii) provide an effective visual presentation of retrieved information.

1.3. Structure of this thesis

This thesis is structured as follows.

Chapter 2 gives an overview of various general topics and methods that are used in this thesis. After a short introduction to machine learning, we present the classifiers and models used in this thesis, as well as the evaluation measures for the classifiers. We then provide more background about the NLP research fields that are important for our work: named entity disambiguation (relevant to Chapters 3 and 4), coreference resolution (relevant to Chapter 3), and sentiment analysis (relevant to Chapter 5).

In **Chapter 3** we start exploring techniques for improved entity retrieval. First, we introduce a method of obtaining relevant context. Next, we describe how we construct a pseudo-ambiguity dataset which allows us to systematically analyze the effectiveness of different types of relevant context. Finally, we conduct several experiments by using classifiers and present our findings, including an investigation of parameters and an error analysis. **Appendix A** gives a detailed overview of all experimental results.

Chapter 4 continues work on entity retrieval as we focus on the problem of little-known entities which are often neglected in IR systems. To address this problem, we present a topic model approach that can identify such entities, and explain the data acquisition process for this identification task. Besides discussing the results of our experiments, we also present an investigation of parameters as well as an error analysis. **Appendix B** lists the number of extracted snippets for the experiments for each entity and gives some more detailed results for some entities.

In **Chapter 5** we go beyond the step of retrieving entities. We present a system that focuses on the identification and presentation of information about entities in a single, self-contained sentence. These sentences contain indicators that allow users to quickly and effectively find information they are looking for. This is done by using a 2-step model that incorporates sentiment classification and a weighing function. For evaluating our experiments we use a crowd-sourcing service, which we describe in more detail including task design and quality control. After discussing the results of our experiments, we provide statistics about the annotators and an error analysis. **Appendix C** presents the top nouns and compound nouns used for computing absolute and relative frequency.

Chapter 6 concludes the work presented in this thesis and summarizes our contributions. It also presents an outlook of possible future work that can be done to extend the techniques presented in this work.

1.4. Publications

Parts of the research described in this thesis were published in the following papers at peer-reviewed international conferences:

- Glaser, A. and Schütze, H. (2012). Automatic generation of short informative sentiment summaries. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL '12)*, pages 276–285 (relevant to Chapter 5)

- Glaser, A. and Kuhn, J. (2014). Exploring the utility of coreference chains for improved identification of personal names. In *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC '14)*, pages 2570–2577 (relevant to Chapter 3)

- Glaser, A. and Kuhn, J. (2016). Named entity disambiguation for little known referents: a topic-based approach. In *Proceedings of the 26th International Conference on Computational Linguistics (COLING '16)*, pages 1481–1492 (relevant to Chapter 4)

The research conducted in these papers was for the most part carried out by myself. The co-authors of these papers acted as advisors with whom I discussed the conception of the architectures presented in the papers. I was the main person responsible for realizing the architectures, including the implementation, carrying out the experiments, evaluating the systems, as well as writing the initial drafts of all papers. My supervisors helped me improve the initial paper drafts.

2. Foundations and background

This chapter describes concepts and techniques that are relevant for this thesis. We start with a short introduction to machine learning and different classifiers and models used in our work. We then introduce the task fields named entity disambiguation, coreference resolution, and sentiment analysis.

2.1. Machine learning

We use different machine learning techniques for all approaches we developed and that we present in Chapter 3, Chapter 4, and Chapter 5. The traditional approach in natural language processing (NLP) has been to develop rule-based systems. A rule-based system consists of a set of hand-written rules, designed to be applied in certain situations regarding the task at hand, including constraints and methods to resolve conflicts between rules. For example, a rule-based sentiment analysis system that determines the polarity (positive or negative) of a given input (e.g., a sentence) could have rules that determine the polarity of compositional phrases that consist of two or more individual phrases of different polarities (e.g., Tan et al. (2015)). The advantage of rule-based systems is that they give users direct control over the system, as rules are easy to create and understand when fixing system problems. They also work well enough if the scope of the task is limited and will not change with new data.

The main problem of rule-based systems is that many applications are far more complex and cannot be modeled adequately with a fixed set of rules. Designing a rule-based system generally starts with the implementation of rules for standard situations. Newly emerging situations and exceptions need to be integrated with their corresponding new rules when they appear. Continuously adding and modifying rules is very time consuming and bloats the code, which results in less comprehensibility.

Maintaining rules is especially problematic in NLP tasks. While many parts of natural languages can be modeled with rules, there are two major problems. First, there are many exceptions, many of which are language-dependent and cannot simply be transferred to other languages. Second, language changes over time, which means rules need to be modified or expanded constantly to capture all the changes. This is easier to achieve for changes in standard language since they are noted down in official documents such as dictionaries and grammar books. It is more challenging when processing non-standard language (e.g., colloquial language and different dialects), as inofficial changes happen more often and are usually not written down. Either way, in both cases much manual work of modifying and adding rules is required and it might become impossible to maintain a rule-based system due to the complexity of rules and the need for new changes all the time.

Machine learning systems, on the other hand, attempt to resolve a given task in a different way, by automatically learning and self-improving with experience—similar to how humans learn. The goal of machine learning is therefore to create computer programs that can learn from some input data to improve their performance. More specifically, Mitchell (1997, p. 2) defines machine learning as:

> "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."

Using sentiment analysis again as an example, the task T would be to classify an input (e.g., a sentence) into positive or negative polarity. The training experience E would be a dataset consisting of sentences with annotated polarity classification, and the performance measure P would be the accuracy of the output.

A machine learning system is not restricted to pre-defined scenarios covered by rules. Instead, each scenario that the system encounters is scored with a prediction score that states how likely the scenario is, and the system can choose and act accordingly. One of the most integral parts of a machine learning system is having a great amount of data to learn from. With more data the system can learn more from the input, i.e., improve more with experience.

2.1.1. Supervised vs. unsupervised learning

There are different learning methods that can be used with machine learning. The two most well-known methods, which we also used in our work, are supervised learning and unsupervised learning.

Supervised learning takes labeled data as input. Labels define to which class (or several classes, in a multi-label classification setting) the data points of the dataset belong to. A supervised system first trains on part of the labeled data (the training set), using certain features. During the testing phase it makes predictions on a test set based on the training. Supervised learning systems can be easily evaluated by comparing the system predictions for the test set with annotations made and checked by human annotators. Such a manually annotated test set is often called a *gold standard*.

Supervised learning systems generally perform well because they learn from annotated data. However, to make good predictions, it is usually necessary to have a large training set. The problem is that manually annotating a large training set can be very expensive. Examples for supervised learning are the three classification methods we use in Chapter 3, Section 3.4.2 and which we describe in more detail in Section 2.1.2: naive Bayes, decision tree learning, and support vector machines. Additional examples include maximum entropy classifiers (which we use in Chapter 5, Section 5.4.2), conditional random fields, and random forests.

Unsupervised learning does not need labeled data. Instead, it takes unlabeled data as input and tries to find patterns and regularities in the data. Based on these patterns, unsupervised systems make predictions for unseen data in the test set. The performance of unsupervised learning may be difficult to determine because there is no manually annotated gold standard. The advantage of unsupervised learning is that it is not necessary to annotate large datasets. Examples of unsupervised learning include clustering methods (e.g., k-means, hierarchical clustering) and certain neural network approaches (e.g., autoencoders, self-organizing maps). The topic modeling approach that we use in Chapter 4, Section 4.3.2 is also a form of unsupervised learning as it does not need labeled data, detects topics based on patterns in the data, and performs a kind of document clustering based on topics (where a document can belong to several topic clusters).

2.1.2. Classification

Classification is a subtask of machine learning that takes a data point X and given a set of classes decides which class X belongs to. Classification is a supervised learning problem and as such needs a training set where data is annotated with the corresponding classes. Based on this training data, a classifier learns which data points belong to which class and can then predict classes for new, unseen input data.

There are two types of classification, binary classification and multiclass classification. In *binary classification*, there are only two classes, for example, “yes” and “no” in email spam classification (e.g., Pantel and Lin (1998)). In *multiclass classification*, the set of classes consists of three or more classes. More than two classes are required for many tasks, such as POS tagging (e.g., Even-Zohar and Roth (2001)) or named entity recognition (e.g., Isozaki and Kazawa (2002)).

Multiclass classification can be decomposed into multiple binary classification tasks (Aly, 2005). This is required for classifiers which can only be applied directly on binary classification problems (e.g., support vector machines use binary classification by default and need to be extended to include multiclass classification). There are two methods for the decomposition. The *one-versus-all* method transforms the classification problem into K binary classification problems, where K is the number of classes. For each class K, a classifier is trained and classifies the input into K or $\overline{K}$, which represents the remaining $K-1$ classes. The *all-versus-all* method trains $\frac{K(K-1)}{2}$ classifiers, each of them on a binary decision between two classes of the training set. In the testing phase, all classifiers are combined and include a voting mechanism. The class with the highest votes will be the output.

We use different classification methods for our approach in Chapter 3, Section 3.4.2, which we describe in more detail in the following subsections: naive Bayes, decision trees, and support vector machines.

Naive Bayes

The *naive Bayes* (NB) classifier, also known as *simple Bayes* or *independence Bayes* (Hand and Yu, 2001), belongs to the class of probabilistic classifiers. Probabilistic classifiers are derived from generative probability models and predict a probability distribution P over a set of classes. The naive Bayes classifier is based on the Bayes theorem, which computes the *posterior* probability of an event B, given an event A:

$$P(B \mid A) = \frac{P(A \mid B)\, P(B)}{P(A)} \tag{2.1}$$

where $P(A \mid B)$ is the probability of A given B and the probabilities of $P(A)$ and $P(B)$ are independent of each other. The following description of the naive Bayes classifier is partly based on (Murty and Devi, 2011, p. 93ff.). In terms of applying the Bayes theorem in a classifier, it calculates the posterior probability of a class C given a set of features $F = \{F_1, ..., F_n\}$ as in the following equation:

$$P(C \mid F_1, ..., F_n) = \frac{P(F_1, ..., F_n \mid C)\, P(C)}{P(F_1, ..., F_n)} \tag{2.2}$$

The denominator $P(F_1, ..., F_n)$ is a constant here since the feature set F is known and does not change. Moreover, it is not dependent on any class C. The numerator $P(F_1, ..., F_n \mid C)\, P(C)$ can be formulated as the joint probability model:

$$\begin{aligned} P(F_1, ..., F_n \mid C)\, P(C) &= P(C, F_1, ..., F_n) \\ &= P(C)\, P(F_1, ..., F_n \mid C) \\ &= P(C)\, P(F_1|C)\, P(F_2, ..., F_n \mid C, F_1) \\ &= P(C)\, P(F_1|C)\, P(F_2 \mid C, F_1)\, P(F_3, ..., F_n \mid C, F_1, F_2) \\ &= \cdots \end{aligned} \tag{2.3}$$

The term "naive" in the name naive Bayes comes from the fact that the classifier assumes that each feature F_i is independent of all other features. This means we can calculate the probability of a feature F_i given a class C without taking the other features F_j (for $i \neq j$) into account:

$$P(F_i \mid C, F_j) = P(F_i \mid C) \tag{2.4}$$

Removing the dependence between features, the joint model from Equation 2.3 then becomes the following:

$$\begin{aligned} P(C)\, P(F_1, ..., F_n \mid C) &\propto P(C)\, P(F_1 \mid C)\, P(F_2 \mid C) \cdots P(F_n \mid C) \\ &= P(C) \prod_{i=1}^{n} P(F_i \mid C) \end{aligned} \tag{2.5}$$

A naive Bayes classifier incorporates this joint model from Equation 2.5 to compute the probability distribution of each feature to all classes. The output depends on a decision rule that is part of the parser. An example of such a decision rule is the *maximum a posterior* (MAP) rule, which simply picks the maximum value from the computed probabilities; the returned result is the predicted class based on the highest probability. This is expressed in Equation 2.6:

$$classlabel = \arg\max_{c} P(C = c) \prod_{i=1}^{n} P(F_i = f_i \mid C = c) \tag{2.6}$$

Despite the naive assumption of independence of features (which is unrealistic since features are usually not completely independent of each other), the classifier works well in practice. Michie et al. (1994) studied the effects of different classification algorithms including naive Bayes and found that the classifier is competitive to other algorithms. Moreover, in cases where the features are indeed mostly independent given the class, naive Bayes performed best in their experiments.

Decision trees

Decision tree learning is a classification method that uses a decision tree. A decision tree is a hierarchical tree-like graph consisting of a root node, internal nodes, leaf nodes, and edges between nodes. The root node and the internal nodes represent tests of an attribute of the input instance, and the values of the attribute are represented by edges (also called branches). Leaf nodes are a representation of the class labels.

Classification with a decision tree starts at the root node and moves down the tree recursively. At every node, the classifier performs a test to obtain the outcome of the attributes given this node. Depending on the value of the attribute for this node, the classifier moves down the subtree to the next node. This step is repeated until the classifier reaches a leaf node. The leaf node yields the class label of the overall decision after taking into account all attributes and values. Decision trees can be binary trees (i.e., have only two decisions such as "yes" and "no") or non-binary (i.e., may have more than two decisions per node).

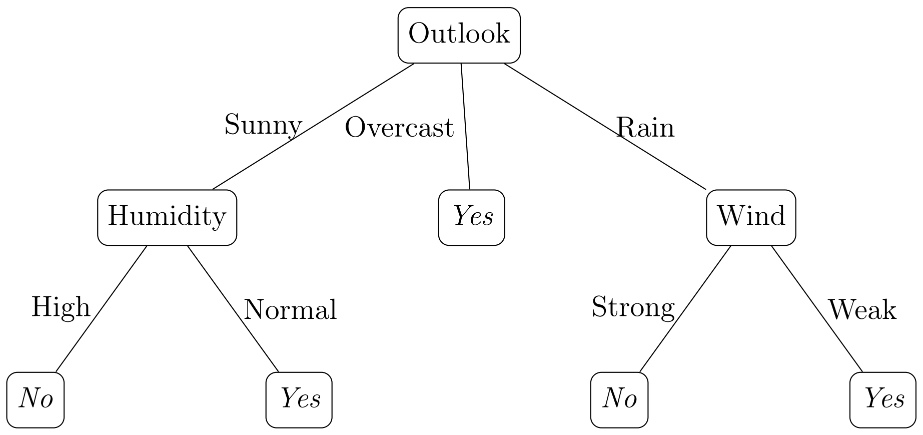


Figure 2.1.: Decision tree for the concept *PlayTennis*, taken from Mitchell (1997).

Figure 2.1 shows a simple non-binary decision tree for the concept *PlayTennis* (Mitchell, 1997). It has three attributes (outlook, humidity, wind) and two class labels (yes, no). Depending on the values of the attributes, an instance would be classified as positive (PlayTennis = yes) or negative (PlayTennis = no).

An advantage of decision trees is that they are human-readable with their tree-like output, which makes it easier to analyze what the classifier is doing during the process. Decision trees are also a good choice when the training data might contain errors or missing attribute values (Mitchell, 1997). If decision trees grow too big they are prone to overfitting on the training data. To overcome this problem, a *pruning* step can be added. The pruning step removes irrelevant nodes and simplifies the tree.

There are different algorithms for decision tree learning, including ID3 (Iterative Dichotomiser 3) (Quinlan, 1983, 1986) and its successor, the popular learning algorithm C4.5 (Quinlan, 1993), which addresses problems of the ID3 and extends the algorithm to resolve them, such as adding a pruning step to prevent overfitting. The C4.5 algorithm is the decision tree algorithm we use in Chapter 3, Section 3.4.2.

Support vector machines

A support vector machine (SVM) is a supervised learning technique that can be used for classification and regression. The goal of a support vector machine is to determine the optimal hyperplane that separates the data points into two classes. We show this with an example classification problem in Figure 2.2. Our data is represented as light red circles in the top right part and dark blue circles in the bottom left part. There are several possible hyperplanes that separate this data. SVMs calculate the distance from each data point to possible hyperplanes and select the one that has the largest margin between the two classes as the optimal hyperplane (solid line). Data points which are located on the margin (dotted line) are called *support vectors*.

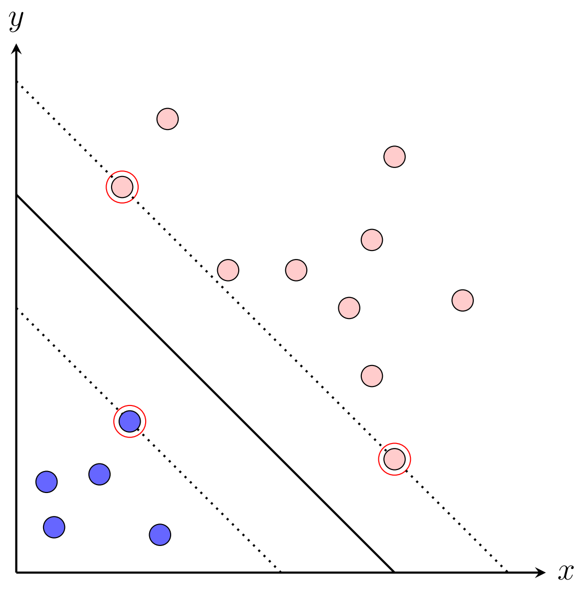


Figure 2.2.: Classification with a SVM. Dotted lines represent possible hyperplanes that separate the two classes of the dataset. The solid line represents the hyperplane with the largest margin. Support vectors on the margin are marked by a red circle around them.

The original SVM algorithm is a linear classifier and works if the training data is linearly separable, i.e., there is a single line that can divide the data points into two classes (Vapnik, 1982). To make it work for non-linear problems (i.e., more lines are needed to separate the data), the SVM algorithm was extended with a kernel trick to make it into a non-linear classifier (Boser et al., 1992; Cortes and Vapnik, 1995). This is done by incorporating a kernel function which transforms the original non-linear feature space into a high dimensional linear feature space.

SVMs are per default binary classifiers. To apply them to multiclass problems, they need to be extended with additional parameters and constraints (e.g., Weston and Watkins (1998); Hsu and Lin (2002); Crammer and Singer (2002); Vural and Dy (2004)). In Chapter 3, Section 3.4.2 we employ a binary SVM.

2.1.3. Topic modeling

Topic models are statistical models that are applied to collections of documents to explore the underlying topics in the collection. In the area of natural language processing, topic models are used to analyze the topics of collections of text documents. The text documents do not need to be preprocessed since topic models work on unstructured text data. Topic models assume that each document in the collection consists of a mixture of topics and each topic consists of similar words that are characteristic for this topic. To illustrate this, Table 2.3 presents four example topics extracted from a text collection. For example, topic 14 consists of words related to music, topic 21 of words related to sports, topic 42 of words related to education, and topic 53 of words related to aerospace[7]. A document in the collection may contain words such as *"school"* and *"students"* as well as *"baseball"* and *"team"*, thus consisting of several topics.

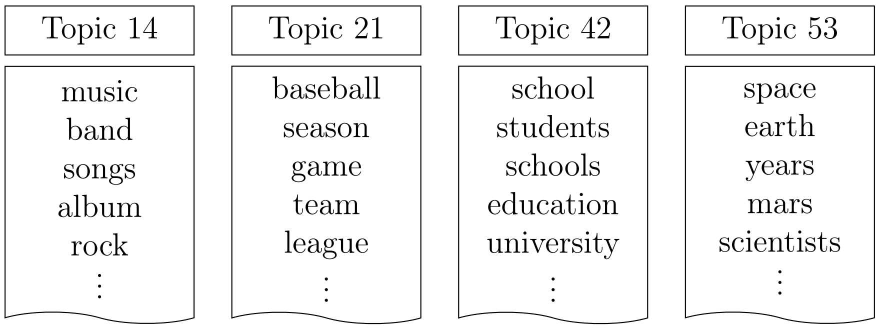


Figure 2.3.: Illustration of four (out of 100) different topics extracted from Wikipedia. Each topic shows the top 5 words.

Different algorithms have been developed to explore text, including latent semantic analysis (LSA) (Landauer and Dutnais, 1997), probabilistic latent semantic analysis

[7]Topic models do not define labels for the extracted topics but when looking through the words in each topic, an approximate label can be given if needed, e.g., for visualization purposes.

(PLSA) (Hofmann, 1999), and latent Dirichlet allocation (LDA) (Blei et al., 2003). In Chapter 4, Section 4.3.2, we use a topic modeling approach based on the LDA method. The following description of the LDA method is partly based on (Blei et al., 2003).

Latent Dirichlet allocation (LDA) is a generative probabilistic model. Blei et al. (2003) describe the idea behind the model as documents being random mixtures over latent topics, and topics consisting of distributions over words. To infer the topics in a new document, the document is generated given the words in the collection in relation to their topics. More formally, the generative process is described as follows, where N is the number of words in a document $\mathbf{w} = (w_1, w_2, ..., w_N)$ and $p(w_n|z_n, \beta)$ is a multinominal probability given the topic z_n. For each document $\mathbf{w}$ in a collection D:

1. Choose $N \sim$ Poission(ξ)
2. Choose $\theta \sim$ Dir(α)
3. For each of the N words w_n:
   a) Choose a topic $z_n \sim$ Multinominal(θ)
   b) Choose a word w_n from $p(w_n|z_n, \beta)$

LDAs can be represented as a graphical model as in Figure 2.4. The model consists of three different levels: (i) the outer level, which consists of the corpus-level parameters α and β, representing the Dirichlet prior on the per-document topic distributions and per-topic word distributions, (ii) the middle level, which consists of the document-level variables θ_d, representing the topic distribution for the document d, (iii) the inner level, which consists of the word-level variables z_n and w_n.

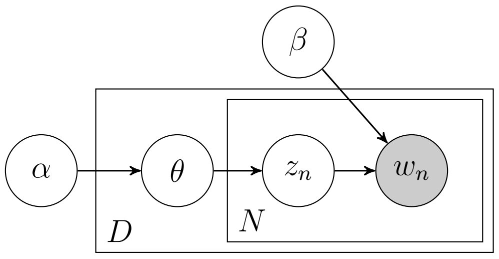


Figure 2.4.: Graphical model representation of LDA.

2.1.4. Evaluation

In this section we describe the evaluation measures that we use to evaluate the classifiers that we described in Section 2.1.2. In the first part, we explain how to compute precision, recall, and F_1 score, which we use as evaluation scores for our results in Chapter 3, Section 3.4.4 and Chapter 4.3.3, Section 4.3.3. In the second part we discuss the difference between micro-average and macro-average in multiclass classification evaluation, which is relevant to our results in Chapter 4.3.3, Section 4.3.3.

Precision, recall, and F_1 score

A standard way for evaluating binary classifiers is using *precision*, *recall*, and *F_1 score*. To calculate these scores, we first need to divide the results of the classification task into four different categories, where *predicted* is the response of the classifier and *reference* is the actual class label in the gold annotated data. These four categories can be presented in a confusion matrix as in Table 2.1.

		Reference	
		Pos	Neg
Pred.	Pos	TP	FP
	Neg	FN	TN

Table 2.1.: Confusion matrix.

- **True positive (TP):** Positive in predicted and reference
- **False positive (FP):** Positive in predicted but negative in reference
- **False negative (FN):** Negative in predicted but positive in reference
- **True negative (TN):** Negative in predicted and reference

Precision (P) is defined as the number of true positives divided by the number of true positives and false positives, i.e., all entities in the predicted response (Equation 2.7). *Recall (R)* is defined as the number of true positives divided by the number of true positives and false negatives, i.e., all entities in the gold reference (Equation 2.8).

$$P = \frac{TP}{TP + FP} \tag{2.7}$$

$$R = \frac{TP}{TP + FN} \tag{2.8}$$

Using only one of these measures does not give a reliable evaluation result as the results can be skewed to favor either precision or recall. For example, for a system that returns many entities in the response, the recall may be higher (since the likelihood of returning correct entities is higher). If the system were to return all entities, it would obtain a perfect recall of 1. However, if all entities are returned there can be more false positives in the results and the precision might be very low.

To get a better understanding of the overall performance of a system, the F_1 *score* (also called *F-score* or *F-measure*) is used. The F_1 score is the harmonic mean of precision and recall:

$$F_1 = 2 \cdot \frac{P \cdot R}{P + R} \tag{2.9}$$

Micro-average vs. macro-average

Precision and recall can only be computed for binary classification decisions, and evaluation of multiclass classification needs to be treated differently. As pointed out in Section 2.1.2, multiclass classification can be decomposed into multiple binary classification tasks. These binary classification results can then be computed and averaged to obtain a general result for the multiclass classification problem. There are two different ways to average results: *micro-averaged* results weight each classification decision equally, i.e., they favor large classes, while *macro-averaged* results weight each class equally, i.e., they show the effectiveness of small classes better (Manning et al., 2008).

In micro-averaging, we compute a special precision score $\mathrm{mi}P$ by summing up all individual true positives of each class $j \in C$ and divide them by the sum of all individual true positives and false negatives (Equation 2.10). The special recall score $\mathrm{mi}R$ is computed the same way, however, instead of false positive we sum up the individual false negatives (Equation 2.11):

$$\mathrm{mi}P = \frac{\sum_{j=1}^{C} TP_j}{\sum_{j=1}^{C}(TP_j + FP_j)} \tag{2.10}$$

$$\mathrm{mi}R = \frac{\sum_{j=1}^{C} TP_j}{\sum_{j=1}^{C}(TP_j + FN_j)} \tag{2.11}$$

In macro-averaging, we first compute the normal precision and recall for each class $j \in C$ as in Equations 2.12 and 2.13:

$$P_j = \frac{TP_j}{TP_j + FP_j} \tag{2.12}$$

$$R_j = \frac{TP_j}{TP_j + FN_j} \tag{2.13}$$

The macro-averaged precision $\mathrm{ma}P$ is then computed by summing up all precision results for each class $j \in C$ divided by the number of classes (Equation 2.14), and the macro-averaged recall $\mathrm{ma}R$ by summing up all recall results divided by the number of classes (Equation 2.15):

$$\mathrm{ma}P = \frac{\sum_{j=1}^{C} P_j}{C} \tag{2.14}$$

$$\mathrm{ma}R = \frac{\sum_{j=1}^{C} R_j}{C} \tag{2.15}$$

The micro-averaged F_1 score is computed as the harmonic mean of the micro-averaged precision and micro-averaged recall (Equation 2.16). The macro-averaged F_1 score is computed as the harmonic mean of the macro-averaged precision and macro-averaged recall (Equation 2.17).

$$\mathrm{mi}F_1 = 2 \cdot \frac{\mathrm{mi}P \cdot \mathrm{mi}R}{\mathrm{mi}P + \mathrm{mi}R} \qquad (2.16) \qquad \mathrm{ma}F_1 = 2 \cdot \frac{\mathrm{ma}P \cdot \mathrm{ma}R}{\mathrm{ma}P + \mathrm{ma}R} \qquad (2.17)$$

The difference between micro-averaging and macro-averaging can best be illustrated with an example. Consider a multiclass classification problem with three classes *{apple, banana, lemon}*. Our test set consists of 71 objects (37 apples, 26 bananas, 8 lemons). The classification results for this test set can be presented in a many-way confusion matrix as the example results show in Table 2.2.

		Reference		
		apple	banana	lemon
Pred.	apple	30	2	0
	banana	3	18	1
	lemon	4	6	7

Table 2.2.: Many-way confusion matrix for a multiclass classification problem (I).

The many-way confusion matrix in Table 2.2 can now be decomposed into n confusion matrices of binary classification problems, where n is the number of classes. Table 2.3 explains where the TP, FP, FN, and TN for each class are located. The diagonal from the top left to the bottom right consists of the TP_j of each class $j \in C$. The sum of the horizontal entries around TP_j are the FP_j of class j, and the sum of the vertical entries around TP_j are the FN_j of class j. The TN_j of class j are the sum of the remaining entries which are not TP_j, FP_j, or FN_j (not listed in the Table 2.3).

		Reference		
		apple	banana	lemon
Pred.	apple	TP_{apple}	FP_{apple} FN_{banana}	FP_{apple} FN_{lemon}
	banana	FP_{banana} FN_{apple}	TP_{banana}	FP_{banana} FN_{lemon}
	lemon	FP_{lemon} FN_{apple}	FP_{lemon} FN_{banana}	TP_{lemon}

Table 2.3.: Many-way confusion matrix for a multiclass classification problem (II).

Decomposing the many-way confusion matrix from Table 2.2 results in the three matrices in Table 2.4, each of them representing a binary classification problem of the sort “one-versus-all”:

		Reference	
		apple	other
Pred.	apple	30	2
	other	7	32

		Reference	
		banana	other
Pred.	banana	18	4
	other	8	32

		Reference	
		lemon	other
Pred.	lemon	7	10
	other	1	53

Table 2.4.: One-versus-all matrices for a multiclass problem.

The numbers from the three matrices in Table 2.4 can now be used to feed Equations 2.10, 2.11, and 2.16 to obtain the micro-averaged results:

$$\mathrm{mi}P = \frac{30+18+7}{30+18+7+2+4+10} = 0.77 \qquad \mathrm{mi}R = \frac{30+18+7}{30+18+7+7+8+1} = 0.77$$

$$\mathrm{mi}F_1 = 2 \cdot \frac{0.77 \cdot 0.77}{0.77+0.77} = 0.77$$

It should be noted that in a multiclass setting, micro-averaged precision and micro-average recall are always the same. To explain this, we first examine the distribution of FPs and FNs in the original many-way confusion matrix in Table 2.3. For each FP there is one FN, which means that $\sum_{j=1}^{C} FP_j = \sum_{j=1}^{C} FN_j$ and with this it follows that Equations 2.10 and 2.11 are the same, i.e., mi$P$ = mi$R$. We can then show that miF_1 = mi$P$ = mi$R$ by substituting mi$R$ by mi$P$ in Equation 2.16:

$$\mathrm{mi}F_1 = 2 \cdot \frac{\mathrm{mi}P \cdot \mathrm{mi}P}{\mathrm{mi}P + \mathrm{mi}P} = 2 \cdot \frac{\mathrm{mi}P^2}{2 \cdot \mathrm{mi}P} = \frac{\mathrm{mi}P^2}{\mathrm{mi}P} = \mathrm{mi}P$$

To obtain the macro-averaged results, we first calculate precision and recall for every class as in Equations 2.12 and 2.13:

$$P_{apple} = \frac{30}{30+2} = 0.94 \qquad P_{banana} = \frac{18}{18+4} = 0.82 \qquad P_{lemon} = \frac{7}{7+10} = 0.41$$

$$R_{apple} = \frac{30}{30+7} = 0.81 \qquad R_{banana} = \frac{18}{18+8} = 0.69 \qquad R_{lemon} = \frac{7}{7+1} = 0.88$$

We then use these intermediate results to calculate the final macro-averaged precision, recall and F$_1$ score using Equations 2.14, 2.15, and 2.17:

$$\mathrm{ma}P = \frac{0.94+0.82+0.41}{3} = 0.72 \qquad \mathrm{ma}R = \frac{0.81+0.69+0.88}{3} = 0.79$$

$$\mathrm{ma}F_1 = 2 \cdot \frac{0.72 \cdot 0.79}{0.72+0.79} = 0.75$$

We can observe that when using macro-averaging, the small class *lemon* has more weight on the final results. The class result itself has a very good recall but the precision is quite low. These values directly affect the final result, which has a lower precision but higher recall compared to the micro-averaged results. The micro-averaged results, on the other hand, favor larger classes and the small class *lemon* does not have a big influence.

2.2. Named entity disambiguation

In this section, we give an overview of the task field of named entity disambiguation (relevant for Chapter 3 and Chapter 4). We also discuss related terms and similar tasks and how they differ from named entity disambiguation.

2.2.1. Introduction to named entity disambiguation

A *named entity* is a unique entity in the real or fictional world (e.g., the person denoted by the name *George W. Bush, 43rd president of the United States*, or *Harry Potter, wizard in a book written by J. K. Rowling*; or the place denoted by the name *Tokyo, capital city in Japan*, or *Nile, river in Egypt*). A *proper name*, also referred to as *proper noun*, is a referring expression in text which is usually employed to refer to a specific named entity (e.g., the referring expression *"George W. Bush"* refers to the 43rd president of the United States), while a *common noun* refers to a generic category (e.g., *"president"*, *"wizard"*, *"city"*, *"river"*). Some sources distinguish between the terms proper name and proper nouns. For example, the Cambridge Grammar of English (Huddleston and Pullum, 2002) defines proper nouns as a single words (e.g., *"John"*, *"Mozart"*, *"Berlin"*) and proper names as containing more than just one proper noun (e.g., *"Bill Clinton"*, *"Mount Fuji"*, *"the Nile"*) or in some cases not containing proper nouns (e.g., *"New Year's Day"*). However, this distinction is not made by everyone and the two terms proper name and proper noun are often used intercheangably. Referring expressions for persons are sometimes also called *personal names* in the literature. In our work we generally use the term *proper name* for referring naming expression consisting of one or several words. However, we use *proper noun* if it refers to the specific tag of the part-of-speech tagger used in Chapter 3.

The term named entity and the task called *named entity recognition (NER)* (e.g., Zhou and Su (2002); McCallum and Li (2003); Ratinov and Roth (2009); Ritter et al. (2011); Lample et al. (2016)) have become popular after being introduced as a task in the Sixth Message Understanding Conference (MUC-6) (Grishman and Sundheim,

Class	Examples
PERSON	Bill Clinton, Albert Gore Sr.
LOCATION	Germany, Atlanta
ORGANIZATION	Microsoft, White House
MISC	Super Bowl, Civil War
MONEY	$ 10 million, $ 200,000
NUMBER	15, sixty, 404-526-5456
ORDINAL	first, one-third, 49th
PERCENT	10 percent, 3 percentage point
DATE	today, last month, 1992, May
TIME	night, 10 a.m., Sunday
DURATION	the last five years
SET	weekly, annually

Table 2.5.: Example classes used in named entity recognition.

1995). NER is concerned with recognizing referring expressions in text and classifying them into different classes. Table 2.5 shows possible classes used in NER, taken from the Stanford Named Entity Recognition tool (Finkel et al., 2005) that we use in Chapter 3, Section 3.4.1, and examples for each class obtained from applying the tool on the English Gigaword corpus. Depending on the design specifications of an NER task or tool there might be more, fewer, or different classes. An overview of the first fifteen years of research in named entity recognition can be found in (Nadeau and Sekine, 2007).

Named entity disambiguation (NED) is the task of identifying referring expressions in text and linking them to their real world or fictional entity. The latter part is what distinguishes NED from NER: while NER classifies found referring expressions into a pre-determined set of classes, NED links referring expressions in text to their corresponding entity, drawn from an unlimited pool of entities.

Referring expressions are often ambiguous and need to be disambiguated before they can be linked to the correct corresponding entity. A human reader can usually resolve the ambiguity given the context of the document. A computer system needs to be able to model this ability. Figure 2.5 shows a text snippet taken from a news article[8]. The two referring expressions *"Michael Jackson"* and *"London"* are both highly ambiguous and can refer to several entities. Apart from the American singer, the proper name *"Michael Jackson"* can also refer to several other people with the same name, but also to songs with the title *"Michael Jackson"*. The expression *"London"* is even more ambiguous: besides referring to multiple cities in the world, it can refer to several songs, films, books, and people with *"London"* as their surname or given name, among other things.

[8]`https://www.africanews.com/2018/06/29/michael-jackson-exhibition-opens-in-london/`

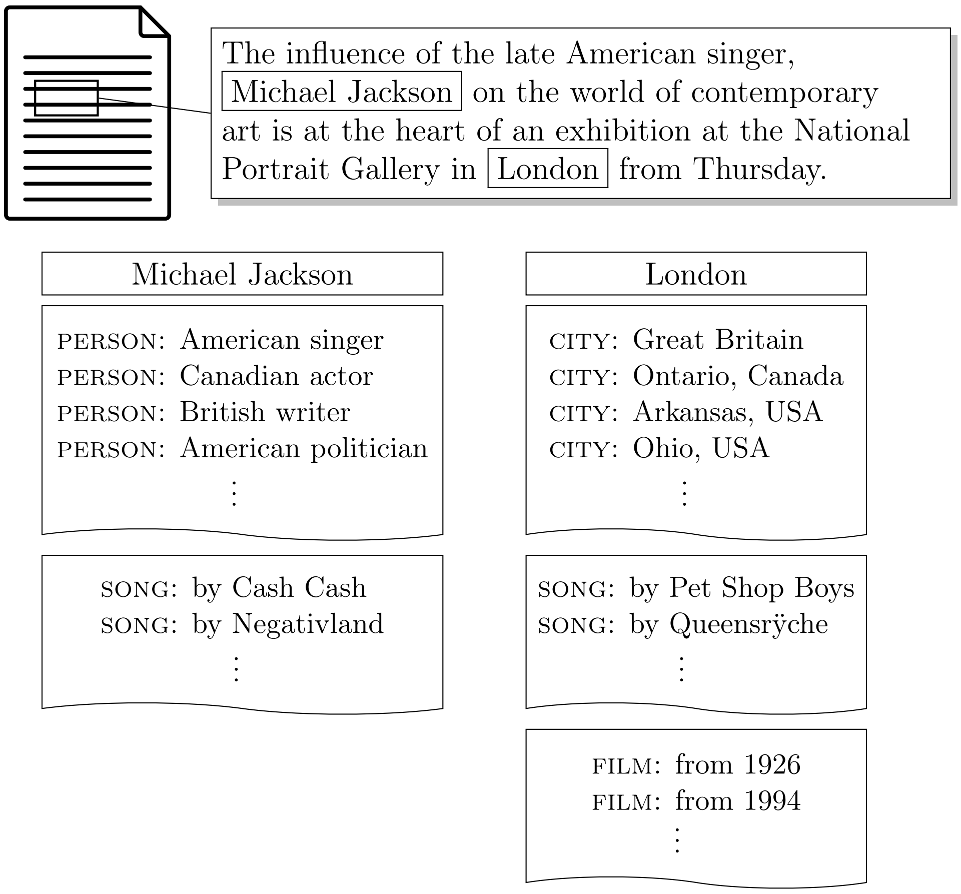


Figure 2.5.: Examples of ambiguous entities.

2.2.2. Naming conventions and approaches

There is a family of tasks closely related to named entity disambiguation. In this section, we give an overview of the different names and how the tasks might differ, as well as discuss the most important approaches.

Some research has been done on *named entity normalization (NEN)* (e.g., Jijkoun et al. (2008); Khalid et al. (2008); Liu et al. (2012)). NEN is defined by these authors as mapping proper names to their unambiguous canonical forms and concepts. This includes disambiguation of proper names and mapping them to their corresponding real world entities (e.g., *"Washington"* can be mapped to the state *"Washington"*, the capital of the US *"Washington, DC"*, or the former president *"George Washington"*). Another part is resolving synonymy, i.e., an entity can be referred to with different proper names (e.g., the concept *"United States of America"* can be referred to with proper names such as *"U.S.", "USA", or "America"*).

A large body of research is concerned with *entity linking (EL)* (e.g., Han et al. (2011); Demartini et al. (2012); Rao et al. (2013); Shen et al. (2015)) or *named entity linking*

(NEL) (e.g., Hachey et al. (2011, 2013); Hajishirzi et al. (2013); Gruetze et al. (2016)). In (named) entity linking, the proper names in a text are linked to their corresponding entities in a knowledge base (KB) (e.g., Wikipedia, DBpedia[9], Freebase[10], or YAGO[11]). There is no clear distinction between EL and NEL in the literature. Sometimes, EL refers to entities in general (including common nouns), while NEL refers to the specific subtask of linking proper names the corresponding knowledge base articles only. In other cases, EL is also concerned with proper names only.

Another term that has become popular in the context of entity linking is *wikification*, also referred to as *disambiguation to Wikipedia (D2W)* (e.g., Mihalcea and Csomai (2007); Milne and Witten (2008); Ratinov et al. (2011); Cheng and Roth (2013); Tsai and Roth (2016)). The goal of wikification is to identify referring entities and concepts (sometimes limited to the most important ones) in text and link them to their corresponding Wikipedia articles. Ambiguities need to be resolved and the correct Wikipedia article chosen. The task differs from NED with respect to the entities being identified. While NED identifies proper names, wikification can also include common nouns.

Knowledge bases are not and will likely never be complete. New entities are emerging every day, for example, people are born and new companies are founded. This means, not every proper name mention can be linked to a corresponding article in a knowledge base. Named entity linking and wikification approaches have different ways for dealing with entities that cannot be linked to an article (sometimes referred to as *out-of-knowledge-base entities* or *emerging entities*). Some approaches assume that the knowledge base is complete (e.g., Cucerzan (2007); Kulkarni et al. (2009); Limaye et al. (2010); Han et al. (2011); Demartini et al. (2012); Han and Sun (2012)). Entities that are not present in the knowledge base are not important enough and thus ignored.

Approaches that do not ignore such entities use different strategies to resolve the problem. Some examples are listed in Table 2.6. In general, there are two strategies: one approach is to return `NIL` if the entity cannot be linked to the knowledge base; another approach is to model a special entity e_{out} which represents out-of-KB entities. Whether or not an entity has a corresponding knowledge base article can be determined in different ways. Some approaches define a threshold that needs to be met or exceeded to be linked to an article. Other approaches use a binary classifier to determine whether a new entity should be linked to a KB entry or not, for example, an SVM classifier (e.g., Zheng et al. (2010); Dredze et al. (2010); Zhang et al. (2011)) or a logistic regression classifier (e.g., Monahan et al. (2011)).

[9] `https://wiki.dbpedia.org/`
[10] `http://www.freebase.com/`
[11] `https://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/yago/`

Work	NIL	e_{out}	Threshold	Classif.
Bunescu and Paşca (2006)		✓	✓	
Zheng et al. (2010)	✓			✓
Dredze et al. (2010)	✓			✓
Zhang et al. (2011)	✓			✓
Monahan et al. (2011)	✓			✓
Pilz and Paaß (2011)		✓	✓	
Gottipati and Jiang (2011)	✓		✓	
Shen et al. (2012)	✓		✓	
Shen et al. (2013)	✓		✓	
Li et al. (2013)	✓		✓	
Guo and Barbosa (2014b)	✓		✓	

Table 2.6.: Strategies for dealing with out-of-knowledge-base entities.

NED shares aspects with *word sense disambiguation (WSD)* (e.g., Yarowsky (1992, 1995); Banerjee and Pedersen (2002); Patwardhan et al. (2003), for a more detailed overview of the task we refer to (Navigli, 2009)). WSD aims at identifying and disambiguating the sense of a word. For example, the word *"bank"* can refer to the financial institution or the land alongside a river or other body of water (among other meanings). Compared to NED, WSD is not restricted to proper names and can include, for example, common nouns. A unified approach that addresses both WSD and NED was done by Moro et al. (2014).

NED also shares aspects with *cross-document coreference resolution (CDCR)* (e.g., Bagga and Baldwin (1998b); Gooi and Allan (2004); Haghighi and Klein (2007); Rao et al. (2010); Singh et al. (2011); Dutta and Weikum (2015)). CDCR takes several documents of a collection and tries to determine which entities across the documents are coreferent, i.e., refer to the same entity. While it is rather unlikely that two or more entities with the same name appear in the same document, it is more likely to happen if several documents are taken. For example, one document could mention *"Michael Jackson, the singer"* while another document could mention *"Michael Jackson, the actor"*. CDCR needs to disambiguate entities to be able to detect coreference correctly.

Approaches used in NED and closely related tasks can be divided into local approaches and global approaches. *Local approaches* disambiguate each mention in a document separately (e.g., Bunescu and Paşca (2006); Mihalcea and Csomai (2007); Ratinov et al. (2011); Lazic et al. (2015); Eshel et al. (2017)). This can be done, for example, by measuring textual similarity between the contexts around the entity and a list of candidate entities or by using other hand-crafted features. *Global approaches* take document coherency into account while disambiguating all mentions of the document simultaneously (e.g., Cucerzan (2007); Milne and Witten (2008); Han and Zhao (2009); Cheng and Roth (2013); Guo and Barbosa (2014a); Pershina et al. (2015); Globerson et al. (2016)). For example, if a document contains the mentions *"Michael Jackson"* and *"Paris"*, and if *"Michael Jackson"* refers to the famous singer then the mention *"Paris"* is most likely to refer to his daughter and not the city in France. This is done, for example, by exploiting the Wikipedia link graph or other relations between entities (e.g., pointwise mutual information (Ratinov et al., 2011) or Jaccard similarity (Guo et al., 2013)). In our work we use a local approach that focuses on the exploration of context features.

There have been several conferences and workshops with specialized tasks for named entity disambiguation and entity linking. The Web People Search Evaluation Campaign[12] (WePS) was hosted three times. In WePS-1 (Artiles et al., 2007) and (Artiles et al., 2009), participants had to disambiguate and cluster entities occuring in web documents. WePS-3 consisted of two tasks: a clustering and attribute extraction task (Artiles et al., 2010) and an online reputation management task (Amigó et al., 2010). In the second task, participants had to analyze company names in twitter posts and determine whether they are related or not.

The Text Analysis Conference[13] (TAC) is conducted by the National Institute of Standards and Technology (NIST). TAC has been hosting knowledge base population (KBP) tracks since 2009 (McNamee et al., 2009; Ji et al., 2010), with the goal of analyzing unstructured text and developing methods to extract information about named entities to populate knowledge bases. The KBP track includes several subtasks such as entity linking (EL), which was later called entity discovery and linking (EDL). In the EL/EDL tasks, named entities need to be linked to their corresponding KB entity (or marked as "NIL" if the entity is not present in the KB).

2.3. Coreference resolution

This section gives a short overview of the task of coreference resolution (relevant for Chapter 3). First, we give an introduction to coreference resolution, including general terminology. Then, we briefly describe approaches and large existing corpora, and present a list of models used in coreference resolution. Finally, we describe the specific model we use in Chapter 3 to obtain automatic cofererence information.

[12]`http://nlp.uned.es/weps/`
[13]`http://www.nist.gov/tac/`

2.3.1. Introduction to coreference resolution

Coreference resolution is the task of determining whether two linguistic expressions refer to the same real world entity or not. These linguistic expressions—usually called *mentions* (e.g., McCallum and Wellner (2003)) or *markables* (e.g., Soon et al. (2001))—are typically noun phrases, such as proper names (e.g., *"Michael Jackson", "Jackson"*), pronouns (e.g., *"he", "him"*), common nouns (e.g., *"singer"*), or complex noun phrases (e.g., *"the famous singer who died in 2009"*). Possessive determiners (e.g., *"his"* in the noun phrase *"his father"*) are usually treated as individual mentions as well even though they are technically not full noun phrases. Besides resolving coreference between entities there are other types of coreference tasks, for example, *bridging* (e.g., Poesio et al. (2004); Markert et al. (2012); Hou et al. (2013, 2014); Rösiger et al. (2018a,b)) and *event coreference* (e.g., Chen and Ji (2009); Chen et al. (2009); Bejan and Harabagiu (2010); Araki et al. (2014); Yang et al. (2015)).

Two mentions are *coreferent* if they both refer to the same real world entity (e.g., *"the famous singer Jackson"* and *"Michael Jackson"*), otherwise they are *disreferent*. Two mentions that are coreferent are connected by a *link*. The first mention in the link is the *antecedent* and the second mention is the *anaphor*. All mentions in a document which refer to the same entity constitute a *coreference chain*. Mentions which are not coreferent with any other mention in the text are called *singletons*.

Consider the following example, taken from the English Gigaword corpus[14]:

(2.1) $\boxed{\text{Ehud Barak}}_1$ told $\boxed{\text{President Clinton}}_2$ of $\boxed{\text{his}}_1$ plans for starting talks with $\boxed{\text{the Palestinians}}_3$. $\boxed{\text{He}}_1$ specifically discussed the 15-month target with $\boxed{\text{Clinton}}_2$, and $\boxed{\text{the president}}_2$ agreed to the urgency.

Example 2.1 shows two sentences annotated with coreference information. These mentions include noun phrases where the head is a proper name (*"Ehud Barak", "President Clinton", "the Palestinians", "Clinton"*), a common noun phrase (*"the president"*), a pronoun (*"he"*), or a possessive determiner (*"his"*).

There are three real world entities in Example 2.1, indicated with different colors and subscript numbers. The proper name *"Ehud Barak"* is coreferent with the possessive determiner *"his"* in the same sentence and the pronoun *"he"* in the next sentence. *"President Clinton"* is coreferent with two mentions in the second sentence, the proper name *"Clinton"* and the common noun phrase *"the president"*. *"The Palestinians"* is not coreferent with any other mention in the example.

[14]See Section 3.4.1 for details on this corpus.

The three coreference chains for this example are the following:

1. Ehud Barak — his — he
2. President Clinton — Clinton — the president
3. the Palestinians

The third chain is considered a singleton as it contains only a single mention (*"the Palestinians"*).

Coreference between mentions is an *equivalence relation*, i.e., it fulfills the following three relations for any mentions m_1, m_2, and m_3:

Reflexive relation: $m_1 = m_1$
Symmetric relation: $m_1 = m_2 \Leftrightarrow m_2 = m_1$
Transitive relation: $(m_1 = m_2 \wedge m_2 = m_3) \Rightarrow m_1 = m_3$

There exist four standard evaluation metrics which can be used to evaluate coreference resolution systems: MUC score (Vilain et al., 1995), B^3 (Bagga and Baldwin, 1998a), CEAF (Luo, 2005), and BLANC (Recasens and Hovy, 2011).

2.3.2. Approaches and models

Coreference resolution has been studied since the 1960s (e.g., Postal (1968); Lakoff (1968); Partee (1970)). In the beginning, automatic coreference resolution was rule-based and used heuristics. For example, Hobbs (1976, 1978) proposes an algorithm traversing parse trees in a certain order and choosing the first noun phrase that matches in gender and number. Other examples for heuristics are the antecedent being in the last n sentences (Charniak, 1972; Klapholz and Lockman, 1975) or treating multiple occurrences of the same pronoun within 1-2 sentences as coreferential (Winograd, 1972).

In the 1990s, the first machine learning approaches for coreference resolution were proposed. For example, McCarthy and Lehnert (1995) use decision trees to resolve coreference. However, due to the lack of annotated corpora not many machine learning based approaches were published. This changed in the mid-1990s, with the advent of several conferences that included subtasks for coreference resolution and provided large datasets annotated with linguistic information including coreference information.

The first and most defining conference was the Message Understanding Conference (MUC), which aimed at evaluating research in information extraction. The subtask of resolving coreference was formally added at MUC-6 (Grishman and Sundheim, 1995; Sundheim, 1995) and continued in MUC-7 (Chinchor, 1998; Hirschman and Chinchor, 1998). They released the first two publicly available corpora which were annotated with coreference information, the MUC-6 corpus and MUC-7 corpus[15]. Both corpora consist of English newswire text.

The Automatic Content Extraction (ACE) Program (Doddington et al., 2004) included coreference resolution tasks many times and produced several corpora annotated with coreference information (ACE-2, ACE03, ACE04, ACE05). Most of these corpora are multilingual (English, Arabic, Chinese) and consist of different types of textual data (including newswire, broadcast news, broadcast conversation, and weblogs).

The OntoNotes corpus (Hovy et al., 2006b; Pradhan et al., 2007), created by the OntoNotes project, is a large corpus annotated with linguistic information, including coreference information. The five released versions consist of various text types (e.g., newswire, broadcast news, broadcast conversations, weblogs) in three languages (English, Chinese, Arabic). OntoNotes was used in the SemEval-2010 Task 1 (Recasens et al., 2010), which addressed coreference resolution in multiple languages, and the CoNLL 2011 (Pradhan et al., 2011) and CoNLL 2012 (Pradhan et al., 2012) shared tasks. For CoNLL they used a specific algorithm to split OntoNotes into training, development, and test partitions (see Pradhan et al. (2011)).

With the advent of these and other publicly available annotated corpora, approaches using machine learning have become quite popular. Table 2.7 lists the most important models that have been developed for coreference resolution. For a detailed overview of coreference resolution models we refer to (Ng, 2010) and (Ng, 2017). An overview of early approaches and machine learning models is given in (Poesio et al., 2016).

Model	Example work
Mention-pair model	Soon et al. (2001), Ng and Cardie (2002)
Mention-ranking model	Yang et al. (2003); Denis and Baldridge (2008)
Entity-mention model	Luo et al. (2004); Yang et al. (2004)
Cluster-ranking model	Rahman and Ng (2009)
Easy-first model	Raghunathan et al. (2010); Lee et al. (2011)
Graph-based model	McCallum and Wellner (2004); Cai and Strube (2010)
Joint models	Denis and Baldridge (2007); Song et al. (2012)
Neural models	Clark and Manning (2016); Lee et al. (2017)

Table 2.7.: Models in coreference resolution.

The model relevant to this thesis is the easy-first model. This model resolves easy coreference decisions in the early stages and delays harder ones until more information is

[15]The corpora mentioned in this section (MUC, ACE, OntoNotes) are distributed by the Linguistic Data Consortium (LDC) at `https://catalog.ldc.upenn.edu/`

available. The first easy-first model for coreference resolution was developed by Raghunathan et al. (2010). They propose a multi-pass sieve model that contains seven layers (sieves) of features, which are processed from easy decisions (exact match) to hard decisions (pronoun match). The cluster output of each sieve is used for the next sieve, which provides more information for later decisions.

Pass	Sieve	R2010	L2011
1	Mention Detection Sieve	✗	✓
2	Discourse Processing Sieve	✗	✓
3	Exact String Match Sieve	✓	✓
4	Relaxed String Match Sieve	✗	✓
5	Precise Constructs Sieve	✓	✓
6-8	Strict Head Matching Sieves A-C	✓	✓
9	Proper Head Word Match Sieve	✗	✓
10	Alias Sieve	✗	✓
11	Relaxed Head Matching Sieve	✓	✓
12	Lexical Chain Sieve	✗	✓
13	Pronouns Sieve	✓	✓

Table 2.8.: Sieves used in multi-sieve coreference resolution systems by Raghunathan et al. (2010) and Lee et al. (2011).

Lee et al. (2011) extend Raghunathan et al. (2010)'s work by adding additional sieves (mostly semantic similarity between mentions), a mention detection sieve at the beginning to filter out unlikely mentions, and a post-processing step for output compatibility reasons. The system participated in the CoNLL 2011 shared task (Pradhan et al., 2011) and outperformed all other systems. The system we use for obtaining coreference information in Chapter 3 implements the multi-pass sieve model described in Raghunathan et al. (2010) with the extensions added in Lee et al. (2011). Table 2.8 shows which sieves are used in the two systems. For a detailed overview of the features we refer to the respective papers. We use the extended model as part of the Stanford CoreNLP system in Chapter 3, Section 3.4.1 for detecting coreference chains in our data.

2.4. Sentiment analysis

This section presents a short overview of the task field of sentiment analysis (relevant for Chapter 5). After a short introduction in general and defining some terms, we give a more detailed overview on different concepts used in sentiment analysis. A more detailed discussion about the task field of sentiment analysis can be found in (Liu, 2015) and (Pang and Lee, 2008).

2.4.1. Introduction to sentiment analysis

Sentiment analysis, also called *opinion mining*, is the task of analyzing opinions expressed in texts. The research field started attracting more interest in the early 2000s (e.g., Tong (2001); Morinaga et al. (2002); Pang et al. (2002); Turney (2002)) and has rapidly expanded since then. Sentiment analysis has become more and more important in the last two decades. It is especially important with the enormous growth of user-generated content on the internet (e.g., reviews, blogs, social networks such as Twitter). Users want to know what others think about a product or service before buying or using it and companies want to know what users think about their products or services. Before the more widespread use of the internet, people had to ask their family or friends for opinions, now they can read many reviews online. Companies can easily gather opinions about their products from reviews, blogs, etc. even without directly asking people about their products.

Analyzing the sentiment or opinion of an expression (e.g., a sentence or a short document such as a tweet) means determining its sentiment polarity. Much previous work has treated sentiment analysis as a binary classification problem, i.e., they classify sentiment into *"positive"* polarity or *"negative"* polarity (e.g., Pang et al. (2002)). Some approaches add *"neutral"* polarity as a third category, which is assigned if the target expression does not convey positive or negative polarity (e.g., Denecke (2008); Melville et al. (2009); Jiang et al. (2011)). Other approaches rate sentiment on a five point scale (*"strongly positive", "weakly positive", "neutral", "weakly negative", "strongly negative"*), which was implemented, for example, in some SemEval tasks that were concerned with sentiment analysis in Twitter (Rosenthal et al., 2015; Nakov et al., 2016; Rosenthal et al., 2017). Sometimes, the category *"mixed"* is added, which is assigned to target expressions that contain mixed polarity (e.g., Jia et al. (2009); O'Hare et al. (2009)). However, identifying the *"mixed"* category can be hard even for human annotators and removing the category can lead to better results (Jia et al., 2009). The following examples illustrate sentences with different polarities:

(2.2) a. *"The picture quality of my new camera is great."* → POSITIVE
b. *"The zoom is very limited and also hard to use."* → NEGATIVE
c. *"I bought the camera yesterday."* → NEUTRAL
d. *"The battery life is great but otherwise the camera is bad."* → MIXED
e. *"This is the best camera I've ever had."* → STRONGLY POSITIVE
f. *"I think the size is fine."* → WEAKLY POSITIVE

Different approaches have been employed to determine sentiment in text. A popular approach is to use a *sentiment lexicon* or *opinion lexicon*, and different methods have been developed for automatic generation of such lexica (e.g., Hatzivassiloglou and McKeown (1997); Turney and Littman (2003); Qiu et al. (2009); Scheible et al. (2010)). A sentiment lexicon is a dictionary that contains sentiment words and their respective polarity. The polarity might be simply encoded as +1 (*"positive"*) and −1 (*"negative"*) (e.g., Hu and Liu (2004c); Kim and Hovy (2004)), but can also take a wider range of values to account for the strength of the polarity. The simplest way to use the sentiment lexicon is to look up all sentiment words that occur in the target expression and sum up all the values for the given words. Negation words such as *"not"* and *"never"* reverse the polarity of the word or phrase they modify. If the final value is positive, the target expression has a positive overall polarity. Conversely, if the final value is negative, the overall polarity is negative.

Some sentiment words convey different polarities, depending on the domain or sometimes even the aspects of a target object. For example, the word *"long"* can be positive in a digital camera domain (e.g., *"The battery life is long"*) but negative in a restaurant domain (e.g., *"The wait for the food was long"*). To resolve this problem, some approaches generate domain-specific sentiment lexica that can capture the differences of different domains or even aspects (e.g., Lu et al. (2011); Huang et al. (2014)).

In the experiment we perform in Chapter 5, the final goal is not determining the polarity of our data. However, it is an important first step to obtain our supporting sentences. For this, we employ a binary sentiment classifier to determine the polarity of sentences, i.e., we restrict the polarity to *"positive"* and *"negative"* (Section 5.4.2).

2.4.2. Subjectivity and objectivity

Subjectivity is related to sentiment, however, the two concepts are different. Liu (2015) defines a *subjective sentence* as a sentence that "expresses some personal feelings, views, or beliefs" and an *objective sentence* as a sentence that "presents some factual information about the world". The following example shows the difference between a subjective and an objective sentence:

(2.3) a. *"I like this camera."* → SUBJECTIVE
b. *"The Canon EOS 200D is a digital camera."* → OBJECTIVE

Sentence 2.3a expresses the speaker's personal feeling about an object (*"this camera"*), here expressed through a verb (*"like"*). The sentence is subjective because the same feeling might not be true for another person. Sentence 2.3b simply states the fact the camera with the name *"Canon EOS 200D"* is a digital camera. Since this is factual information the sentence is an objective sentence.

In most cases, subjective sentences contain sentiment. However, a subjective sentence does not necessarily contain any sentiment. Consider the following examples:

(2.4) a. *"I think I like this camera."* → SUBJECTIVE + SENTIMENT
b. *"I believe he owns this camera."* → SUBJECTIVE + NO SENTIMENT

Both sentences 2.4a and 2.4b are subjective sentences as they state the speaker's feeling or belief. Sentence 2.4a does contain the speaker's sentiment about something (expressed by the verb *"like"*). However, sentence 2.4b does not express any sentiment.

Likewise, an objective sentence is not always free from sentiment but can contain implicit sentiment (Zhang and Liu, 2011):

(2.5) a. *"The camera broke right after I bought it."* → OBJECTIVE + SENTIMENT
b. *"A valley has formed in the mattress I bought last month."* → OBJECTIVE + SENTIMENT

Sentence 2.5a states a fact about the condition of a camera. Since breaking immediately after the purchase is not something someone would expect about a good product, this sentence expresses an implicit negative sentiment. The same applies to sentence 2.5b, where a deformation of a new mattress is not expected to happen that fast.

Subjectivity and objectivity of sentences and how to classify them has been studied in previous work (e.g., Riloff and Wiebe (2003); Yu and Hatzivassiloglou (2003); Wiebe and Riloff (2005); Raaijmakers et al. (2008); Banea et al. (2008); Benamara et al. (2011); Rustamov et al. (2013)).

In our work in Chapter 5, we do not make a formal distinction between subjective and objective sentences. We assume that good supporting sentences can be found in both subjective and objective sentences. However, we require all sentences to express the sentiment of the author, either explicit or implicit.

2.4.3. Granularity levels of analysis

Sentiment analysis can be applied to different granularity levels: from a coarse-grained level (document level), over medium-grained levels (sentence level and phrase level), up to a fine-grained level (aspect/entity level).

Document level

In document level sentiment analysis, the entire document (e.g., a customer review) is taken as the input and the result is the polarity of the entire document (e.g., Pang et al. (2002); Turney (2002); Beineke et al. (2004); Yessenalina et al. (2010); Maas et al.

(2011); Moraes et al. (2013); Yang et al. (2016)). It is assumed that the document does express sentiment (positive or negative) and that the sentiment is about a single object (Liu, 2010).

To determine the overall polarity of a document, usually all sentences are taken into account. However, some approaches discard objective sentences and determine the overall polarity based on the subjective sentences only (Pang and Lee, 2004). The document is then labeled with the polarity that is more present in the document. While this approach determines the overall opinion of a document, it does not actually differentiate between individual sentences. For example, if a document consists of 60% positive sentiment and 40% negative sentiment, the overall polarity of the document will be positive, even though there is a considerable amount of negative sentiment present.

Sentence/phrase level

Sentence level sentiment analysis determines the polarity of a sentence (e.g., Yu and Hatzivassiloglou (2003); Kim and Hovy (2004); Meena and Prabhakar (2007); Täckström and McDonald (2011); Socher et al. (2011); dos Santos and Gatti (2014); Kim (2014)). This allows for a more fine-grained analysis of what an opinion holder is thinking about an object (e.g., a product). For example, in many positive product reviews there is still at least one negative sentence, and vice versa. Sentence level sentiment analysis can identify such sentences and provide a better understanding of what an opinion holder thinks is positive and negative.

While sentence level sentiment analysis provides a more fine-grained analysis compared to document level sentiment analysis, it is not fine-grained enough in certain situations. Sentence level sentiment analysis generally assumes that a sentence expresses one sentiment (positive, neutral, and sometimes neutral or "no opinion"). However, it is possible that a sentence contains different types of sentiment as in the following example:

(2.6) *"I like the zoom of this camera, but the battery life is too short."*

The first part of example 2.6 expresses positive sentiment, while the second part expresses negative sentiment. No matter which sentiment is taken as the final sentiment for the entire sentence, it will neglect the other sentiment. Another issue that can be seen in this sentence is that not every sentence is about a single object or single aspect of an object. This sentence evaluates two different aspects (*"the zoom of this camera"* and *"the battery life"*), but analyzing the sentence polarity only will not give more detailed information about the aspects that are evaluated.

Even more fine-grained is phrase level sentiment analysis (e.g., Nasukawa and Yi (2003); Wilson et al. (2005, 2009); Agarwal et al. (2009); Yessenalina and Cardie (2011);

Severyn and Moschitti (2015)). Work on phrase level sentiment analysis attempts to determine the polarity of individual phrases, for example, subjective phrases or in a more specific setting subject phrases. The polarity of individual phrases in a sentence can differ from the polarity of the sentence they occur in as can be seen in the following example, taken from (Agarwal et al., 2009):

(2.7) *"The robber entered the store but his efforts were crushed when the police arrived on time."*

The underlined phrase in sentence 2.7 expresses a negative polarity. However, the sentiment of the entire sentence is positive.

Aspect/entity level

Aspect level sentiment analysis, also called entity level sentiment analysis or feature level sentiment analysis, is the most fine-grained type of analysis and has been studied in detail in the literature (e.g., Hu and Liu (2004c); Popescu and Etzioni (2005); Zhuang et al. (2006); Wang and Wang (2008); Ding et al. (2008); Zhang et al. (2010); Yu et al. (2011); Kim et al. (2013); Wang et al. (2016)). It is very close to phrase level sentiment analysis but not exactly the same. While phrase level sentiment analysis looks at phrases in general, aspect level sentiment analysis aims to examine the actual opinion, consisting of a target (e.g., an object, or an aspect of an object) and a sentiment about it (positive, negative, neutral) (Liu, 2015). With this, aspect level sentiment analysis can identify different aspects and different sentiments in a single sentence:

(2.8) *"The food in this restaurant is very good, but the service is horrible."*

Sentence 2.8 is a typical sentence from a restaurant review. It contains two aspects of a restaurant (*"the food"* and *"the service"*) with different polarities: the first part about the food is positive, while the second part about the service is negative.

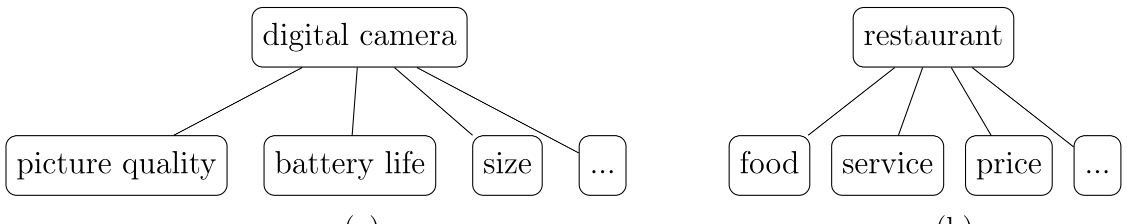


Figure 2.6.: Examples of aspects of two different objects of interest—a product (digital camera) and a service (restaurant).

Figure 2.6 shows two different target objects. A digital camera may have aspects such as *"picture quality"*, *"battery life"*, or *"size"*. A restaurant may have aspects such as *"food"*, *"service"*, or *"price"*. For aspect level sentiment analysis, it is important to identify these aspects before determining the sentiment.

Aspect level sentiment analysis consists of several subtasks. First, the target object (main entity) of the document needs to be extracted (e.g., *"camera"* or *"restaurant"* in a consumer review). After this, the aspects of the target object are extracted from the document. Liu (2015) describes four approaches for aspect extraction: (i) identifying frequent nouns phrases (e.g., Ku et al. (2006)), (ii) using syntactic relations such as syntactic dependencies of a dependency graph (e.g., Zhuang et al. (2006), (iii) applying a supervised learning method such as conditional random fields (e.g., Jakob and Gurevych (2010)), or (iv) using topic models (e.g., Branavan et al. (2009)). If a document contains several possible opinion holders, they also need to be identified. Then, the sentiment for these aspects is determined (i.e., positive, negative, or neutral). Some approaches aggregate all opinions about specific aspects or even rank them for a better presentation of the results.

For our experiments in Chapter 5, we perform sentiment classification on the sentence level (Section 5.4.2) as a first step to determine the polarity of our supporting sentences.

3. Informed context selection for named entity disambiguation

3.1. Motivation

In this chapter we discuss how to improve the overall accuracy of name identification and disambiguation by making a linguistically informed choice of context[16]. Context plays an important role in the task of named entity disambiguation. If an ambiguous proper name appears with no or only little context, or if the context does not provide any distinct words or features, then it is hard or even impossible to disambiguate the proper name. Consider the following example contexts:

(3.1) a. *"Michael Jackson was here 20 years ago."*
b. *"Michael Jackson, the American singer, was here 20 years ago."*
c. *"Michael Jackson performed his moonwalk dance at yesterday's concert."*
d. *"Michael Jackson died on June 25 at the age of 50."*

The context provided in sentence 3.1a is not very informative. By only reading this sentence with no further information, it is not clear who of the many people with the name *"Michael Jackson"* is mentioned here. In fact, depending on where and when the statement is made, it can apply to every *"Michael Jackson"* who is at least 20 years old. To know who the entity in question is, additional context is necessary.

Sentence 3.1b adds the appositive noun phrase *"the American singer"* to the proper name *"Michael Jackson"*. With this description we can narrow down the pool of possible candidates for the entity to all people with the name *"Michael Jackson"* that are American and a singer. In sentence 3.1c, the words *"his moonwalk dance"* and *"concert"* describe the entity in more detail. The context in 3.1d contains information about when the entity died and the age of death. This is very helpful information because the likelihood of two or more people with the same name dying on the exact same day at the exact same age is very low. Even without having world knowledge, a classifier can

[16]The approach presented in this chapter was published in (Glaser and Kuhn, 2014).

make use of descriptive words and phrases like in sentences 3.1b–3.1d and learn how to disambiguate an entity with them.

The classic approach in NED is to use a bag-of-words (BOW) model (e.g., Bagga and Baldwin (1998b); Fleischman and Hovy (2004); Gooi and Allan (2004); Pedersen et al. (2006)), in which text is represented as an unordered list of its words and their frequencies in the text. This can be implemented as vectors, where each position in the vector represents a word in the vocabulary and each value is the number of occurrences of the word in the text. The model then computes the similarity between a context vector from the test set and the context vectors from the training set, and predicts the entity with the highest context similarity as the correct entity. While the model does not take grammatical or semantic information into account, it can effectively capture general topics of the texts (Jurafsky and Martin, 2009).

Standard approaches use different context sizes around the entity that needs to be identified. Some approaches use the entire document (e.g., Bollegala et al. (2006); Ikeda et al. (2009); Hoffart et al. (2011)) to obtain the maximum of information. Other approaches assume that the most important information is close to the entity and use a fixed window of n words or n-grams around the entity (e.g., Pedersen et al. (2005); Bunescu and Paşca (2006); Nguyen and Cao (2008); Ikeda et al. (2009); Han et al. (2011); Li et al. (2013)).

How important context is and especially the choice of which context to use for the disambiguation task is illustrated in the following example taken from a news article[17]:

(3.2) “Hollywood hunk Matthew McConaughey wants to name his unborn son after his favorite **beer**. The Fool’s Gold star—who is expecting his first baby with girlfriend Camila Alves—is planning to pay tribute to his beloved **alcoholic beverage**, just like his brother did.

A source close to the actor said: ‘Matthew’s older brother Michael named his second son Miller Lyte because he loved the **beer** so much. And Matthew loved the name so much he really wants to name his son after his favorite **beer**. He is thinking of going for Bud after **Budweiser beer**.’

It wouldn’t be the first child of a star with an unusual name, other prominent parents are Chris Martin or **<u>Michael Jackson</u>**. Martin and his wife Gwyneth Paltrow named their daughter Apple. Jackson used a family name that goes back four generations. He called his sons Prince Michael I and Prince Michael II.

However, Brazilian model Camila is less than impressed with McConaughey’s choice of name. The source added: ‘Camila is pretty old-fashioned. She hates the name and won’t let Matthew push her into naming their baby after his favorite **beer**.’ ”

[17]`http://www.stuff.co.nz/entertainment/308564/McConaughey-to-name-baby-after-beer`

Example 3.2 shows a text document that contains the proper name *"Michael Jackson"*, which refers to the famous American singer. Identifying the correct real world entity is difficult in this case because most content of the document is not relevant to the entity and contains information that can be misleading for a classifier. Approaches that use the entire document or a small fixed window of words or sentences around the entity will have problems with correctly identifying in such cases, as described below.

Using the entire document. The entire document gives the maximum of information. However, if a document is not about the entity in question alone but also about other entities besides the one we are interested in (which is usually the case, for example, even lexicon articles contain other entities). Using the entire document as context might be possible if the documents are rather small and very focused on the entity that needs to be identified. However, if documents are larger and contain information about many different entities, this approach does not work well because it provides too much information that is not relevant for the target entity. A classifier using this approach would most likely classify the *"Michael Jackson"* in Example 3.2 as the beer expert because of several mentions of *"beer"* and *"alcoholic beverage"* throughout the entire document.

Using the sentence in which the entity occurs. The idea behind using the sentence in which the entity occurs, is that this sentence often contains useful information describing the entity. However, this is not always the case. For example, the sentence that contains *"Michael Jackson"* in Example 3.2 does not provide useful information about this entity. The only information it provides is that this *"Michael Jackson"* gave his child an unusual name, but this information is hard to learn for a classifier as it requires world knowledge (e.g., *"Which Michael Jackson has a child with an unusual name?"* and *"What is an unusual name?"*). As can be seen in this example, a context window of one sentence might contain useless information that does not help identify the entity.

Using the sentence the entity occurs in plus the previous/next sentence. More context can provide more helpful information. Very often, the sentences around the entity contain further information. However, this is not always the case as can be seen in Example 3.2. The next sentence only adds more information about another person who gave their child an unusal name. The previous sentence also contains information about a different entity which in this case might cause a misclassifiction because of the keyword *"beer"*.

Using a window of n words or n-grams around the entity. Some approaches do not use whole sentences as context but use a fixed window of words or n-grams around an entity (Pedersen et al., 2005; Nguyen and Cao, 2008; Ikeda et al., 2009; Han et al., 2011; Li et al., 2013). These approaches have to deal with similar problems as approaches that use sentences as context window, i.e., the words in the context window might not be useful or might contain information that can lead to misclassifications. It is also difficult to set n in any principled way.

To overcome the problems of these standard ways of determining context (i.e., using the entire document or a fixed window around the entity), we define a new type of context, which we call *r-context* (*relevant context*). Instead of taking all information without a filter or restricting the context with an arbitrary number of words or sentences, we use coreference information about the entities in the text to obtain only the parts of the document which contain relevant information about the entity that needs to be identified. We first identify coreferent expressions in the document and then combine all sentences which contain coreferent mentions to one single context. Consequently, our first research question is:

Research Question 1: How helpful is it for named entity disambiguation to expand context in a linguistically informed way by adding the context of expressions from the same document that are (automatically) identified as coreferent?

To answer this question, we use an experimental set-up that allows us to systematically explore the effects of different sizes and types of context. We apply different filters to obtain different types of coreference-based context of different sizes. Our exploratory study focusses on the following three hypotheses:

Hypothesis 1: A larger window around mentions of names provides more reliable feature information than a smaller window.

Hypothesis 2: A structurally informed choice of the context window improves the disambiguation.

Hypothesis 3: Identifying coreference chains of entities provides relevant context.

We use two kinds of annotations for training and evaluation: (i) referent identification of proper names and (ii) coreference information. However, there is a lack of large enough datasets annotated with both types of information and hand-labeling a dataset of a decent size is expensive. To solve this problem we created a pseudo-ambiguity dataset that gives us an analytical advantage: it allows us to experiment with enough data to study different corpus sizes as training material without the need to annotate a large dataset by hand. Pseudo-ambiguity has been introduced for word sense disambiguation

(Schütze, 1992; Gale et al., 1992) and has later been used for disambiguating named entities (e.g., Mann and Yarowsky (2003); Pedersen et al. (2005)). For a comparison of pseudo-ambiguity data and real data, we annotate a small dataset by hand with information about ambiguous entities. Our results on both the pseudo data and the real data show that using coreference-based context performs better than using a fixed window of context around the entity.

In summary, we make the following contributions in this chapter:

- We show that our approach of using structurally informed choice of context generally helps improve the disambiguation of proper names.
- We describe the process for creating a large pseudo-ambiguity dataset that can be used for experiments, and show that the results are comparable to those performed on real data.
- We perform a systematic study with different parameters (corpus size, corpus creation method, context types) on the pseudo-ambiguity dataset and present detailed results which show the effects of these parameters.

The rest of this chapter is structured as follows. We define a technical notion of relevant context (*r-context*) and how it is created in Section 3.2. In Section 3.3 we discuss why we use a pseudo-ambiguity dataset, how it is created, and how we select names for the experiments. We describe our experiments and discuss the results in Section 3.4. The discussion includes an analysis of the different parameters we used. At the end of this chapter we show an error analysis. We present related work in Section 3.5 and conclude with a summary in Section 3.6.

3.2. r-context

As shown in Section 3.1, the choice of context that is being considered is important to disambiguate an entity correctly. Choosing the entire document as context provides noisy information and taking a fixed window around the entity can result in context that is not relevant to the entity at all.

In our work we systematically explore different types and sizes of contexts which are relevant to the person that needs to be disambiguated. We call these contexts *r-context* (*relevant context*). The contexts are selected by automatically extracting only those sentences from a document which are relevant to the person. We define a sentence to be relevant if the entity is directly mentioned in the sentence. A mention can be a proper noun, common noun, or pronoun.

Extracting relevant sentences is not a trivial task. Sentences that mention the entity with a proper noun can be extracted with a string matching algorithm to some extent. However, such an algorithm has problems if there is more than one person with the same name in a document, e.g., *"Bill Clinton"* and *"Hillary Clinton"*. A simple string matching algorithm would extract all sentences that contain the last name *"Clinton"* without differentiating between the two persons. A more sophisticated approach to determine which entities really belong together is using coreference resolution. In coreference resolution, more features can be used to determine entities, for example, the distance between entities or information about gender.

Example 3.3 shows a short document[18] about the entity *"Ehud Barak, the Israeli Prime Minister"*. All mentions that are coreferent with this entity are marked with boxes and the corresponding part-of-speech (POS) tag.

(3.3) "Israelis overwhelmingly elected $\boxed{\textbf{Ehud Barak}}_{[\text{NNP}]}$, the $\boxed{\text{Labor Party leader}}_{[\text{NN}]}$, as $\boxed{\text{prime minister}}_{[\text{NN}]}$ on Monday. In a complete reshuffling of the political deck, Israelis also remade their parliament. $\boxed{\text{Barak}}_{[\text{NNP}]}$ has the option to form a government without any ultra-Orthdox representation. By 3 a.m. Tuesday, $\boxed{\text{he}}_{[\text{PRP}]}$ was leading with 57 percent of the vote to Netanyahu's 42.8 percent. Within 35 minutes of the first television exit polls, Netanyahu not only conceded defeat, but stepped down as the leader of the conservative Likud Party. The new $\boxed{\text{prime minister}}_{[\text{NN}]}$ now has 45 days in which to form $\boxed{\text{his}}_{[\text{PRP\\$}]}$ government."

Table 3.1 lists all references of *"Ehud Barak, the Israeli Prime Minister"* from the document in example 3.3 with their respective POS tags. The entity can be referenced with a proper noun (NNP), e.g., the full name *"Ehud Barak"* or only the last name *"Barak"*. Another very common reference is the use of pronouns. These can be personal pronouns (PRP), e.g., *"he"*, or possessive pronouns or determiners (PRP$), e.g., *"his"*. Entities can also be referred to with a common noun (NN), e.g., *"Labor Party leader"* or *"prime minister"*.

POS	Examples
NNP	Ehud Barak, Barak
NN	Labor Party leader, prime minister
PRP	he
PRP$	his

Table 3.1.: Name mentions in the r-context.

[18]Taken from the English Gigaword corpus; shortened version for simplification.

A coreference resolution system can determine these references and group all coreferent mentions in one coreference chain. The coreference chain for this example is shown in 3.4. The subscript indicates the sentence number in which the mention occurs.

(3.4) $\boxed{\text{Ehud Barak}}_{S1}—\boxed{\text{Labor Party leader}}_{S1}—\boxed{\text{prime minister}}_{S1}—\boxed{\text{Barak}}_{S3}—$

$—\boxed{\text{he}}_{S4}—\boxed{\text{prime minister}}_{S6}—\boxed{\text{his}}_{S6}$

We use the coreference information obtained from a coreference resolution system to obtain r-context. For each entity e, we perform the following steps:

1. We extract the *base context* for e. The base context is defined as the sentence in which e occurs.
2. We determine all coreferent mentions $m_1 \dots m_n$ of e in the document with a coreference resolution system. All coreferent mentions for e constitute the coreference chain for e.
3. For every mention m_i in the coreference chain, we extract the sentence in which m_i occurs as a potential candidate sentence to be included in our r-context.
4. The extracted sentences will then be added to the base context depending on one of the following four filters:
   - **NNP:**
     Use only sentences that contain the entity as a proper noun.
   - **NNP+NN:**
     Use sentences that contain the entity as a proper noun or common noun.
   - **NNP+PRP:**
     Use sentences that contain the entity as a proper noun or pronoun. We do not make a distinction between personal pronouns (PRP) and possessive pronouns (PRP$) but merge all pronouns into one category (PRP).
   - **NNP+NN+PRP:**
     Use all sentences that contain the entity.

Table 3.2 illustrates different context types for identifying the proper name *"Ehud Barak"* in the first sentence of example 3.3. The first two rows present context obtained by fixed window approaches around the entity. The first row shows the base context, i.e., only the sentence in which the entity occurs. The second row adds the previous and next sentence to the context. Since the entity is in the first sentence in this example, no previous sentence is added to the context. The next sentence is added to the context. However, it is not about the entity, so it does not provide relevant information.

Fixed window	Context
BASIS	Israelis overwhelmingly elected **Ehud Barak**, the *Labor Party leader*, as prime minister on Monday.
BASIS +PREVIOUS/ NEXT SENTENCE	Israelis overwhelmingly elected **Ehud Barak**, the *Labor Party leader*, as prime minister on Monday. In a complete reshuffling of the political deck, Israelis also remade their parliament.
r-context filter	**Context**
NNP	Israelis overwhelmingly elected **Ehud Barak**, the *Labor Party leader*, as prime minister on Monday. *Barak* has the option to form a government without any ultra-Orthdox representation.
NNP+NN	Israelis overwhelmingly elected **Ehud Barak**, the *Labor Party leader*, as prime minister on Monday. *Barak* has the option to form a government without any ultra-Orthdox representation. The new *prime minister* now has 45 days in which to form a government.
NNP+PRP	Israelis overwhelmingly elected **Ehud Barak**, the *Labor Party leader*, as prime minister on Monday. *Barak* has the option to form a government without any ultra-Orthdox representation. By 3 a.m. Tuesday, *he* was leading with 57 percent of the vote to Netanyahu's 42.8 percent.
NNP+NN+PRP	Israelis overwhelmingly elected **Ehud Barak**, the *Labor Party leader*, as prime minister on Monday. *Barak* has the option to form a government without any ultra-Orthdox representation. By 3 a.m. Tuesday, *he* was leading with 57 percent of the vote to Netanyahu's 42.8 percent. The new *prime minister* now has 45 days in which to form a government.

Table 3.2.: Examples of different context types with fixed sentence windows (first two rows) and our system's filter methods (last four rows). The entity that needs to be disambiguated is marked in bold. Mentions that are coreferent with this entity are marked in italic.

The last four rows in Table 3.2 show the r-context we created with our filter methods and illustrates different context types depending on which types of POS tags are included in the filter method. Mentions which are coreferent with the original entity are marked in italic. If the entity is mentioned several times in a sentence (e.g., as a proper noun and a common noun like in the first sentence), this sentence will be used only once.

We test different filters for two reasons. First, we want to investigate the effect of different context types and whether every sentence that contains the entity adds useful information to the context. Second, since we do not have gold information for coreferent entities but determine coreference automatically, it is a way to measure the quality of the coreference information. A coreference chain is not completely reliable and can contain wrongly classified entities. Classifying proper nouns correctly is easier because the string matching feature is quite reliable. However, pronouns are always more difficult to identify correctly. Adding a sentence to the context that is about another entity, adds more noise to the context.

3.3. Pseudo-ambiguity dataset

This section consists of three parts. In the first part, we present existing datasets for the NED task (both created with real data and pseudo-ambiguity data). We explain why they are not suitable for our task and motivate the reason for creating a new pseudo-ambiguity dataset. In the second part, we describe the difference between a standard dataset and a pseudo-ambiguity dataset, as well as the creation process. In the last part, we discuss how we select names for our pseudo-ambiguity dataset.

3.3.1. Motivation

A suitable dataset for our task must contain two important types of annotation. First, proper nouns need to be referenced with their corresponding entity to perform the actual classification task. Second, since the creation of r-context relies on the availability of coreference chains, the data also needs to be annotated with coreference information. There are several datasets with coreference annotations available, for example, MUC (Grishman and Sundheim, 1995), ACE (Doddington et al., 2004), and OntoNotes[19]. These datasets are also annotated with other linguistic information such as part-of-speech tags or parses. However, they do not contain annotations for the real world referents of proper names. The annotation, including the coreference information, was annotated manually for these datasets. Such gold labels provide more accurate information, but when it comes to real applications, gold labels are usually not available. For a realistic exploration of a task it is better to use automatically predicted annotations because gold labels might provide overly positive results which are not achievable with automatically predicted annotations.

On the other hand, there are some available datasets that are annotated and used for named entity disambiguation, but they usually do not contain coreference information. The only dataset we could find that contained both coreference information and real world references was created by Ratinov et al. (2011) by taking a subset of the ACE corpus and having human annotators on Amazon Mechanical Turk[20] link mentions to Wikipedia articles. However, the dataset is quite small. It consists of 257 entities of which 255 are linked to distinct Wikipedia articles, which leaves almost no ambiguity to resolve.

Other available datasets for named entity disambiguation use subsets of Wikipedia (Cucerzan, 2007; Ratinov et al., 2011), the AQUAINT Corpus[21] (Milne and Witten,

[19]Detailed information on OntoNotes can be found in Section 3.4.1.
[20]A description of Amazon Mechanical Turk can be found in Chapter 5.3.2.
[21]`https://catalog.ldc.upenn.edu/LDC2002T31`

2008), and the CoNLL 2003 corpus for NER without coreference information (Hoffart et al., 2011). Cucerzan (2007) downloaded news articles from `msnbc.com`. Röder et al. (2014) created three new corpora for named entity disambiguation (100 German news articles from `news.de`, 128 articles from Reuters-21578[22], 500 sentences from 1,457 RSS feeds). However, these datasets were published more than a year after we finished work on our task. Another dataset called KORE50 (Hoffart et al., 2012) comprises 50 difficult sentences based on several criteria such as short context, high density of mentions, and highly ambiguous mentions. In general, these datasets have several problems that make them not suitable for our task. One problem is that these datasets are usually small. We show in our preliminary results that a small dataset is not enough to make reliable predictions. Another problem is that many of these datasets do not contain many ambiguous entities.

There are two corpora that provide a higher density of ambiguous names. The John Smith Corpus (Bagga and Baldwin, 1998b) comprises 197 documents that contain the name *"John Smith"* which refers to 35 different entities. However, since it only contains documents about one name it is not sufficient for an exhaustive evaluation.

The Kulkarni Name Corpus[23] (Kulkarni, 2006) contains documents about 5 ambiguous person names which refer to 15 unique entities (2-4 entities for each name). Overall, the corpus contains 1,375 documents. However, the author limits the context around each entity to approximately 100 words (with the entity in the middle). This makes it not suitable for our task which exploits gathering information about entities from the entire document, not just a context window around the entity.

The Text Analysis Conference (see Chapter 2, Section 2.2) provides datasets for each track, including the knowledge base population track (KBP) that has the entity linking subtask. However, the TAC-KBP data is for the most part only available to participants of the track. The data used in the years 2009-2011 was later made available on LDC[24] but only one year after we finished work on this task.

The Web People Search Evaluation Campaign (see Chapter 2, Section 2.2) provides freely available datasets for all their three tasks. Given the nature of their task, these datasets look very relevant for our own task. However, there are some problems with them. The datasets were created by taking 30 random names from 3 different sources (Wikipedia, ACL 2006 authors, US Census; 10 names for each source) and downloading the top 100 webpages in a search engine result for each of the names in the first two tasks (Artiles et al., 2007, 2009). In the third task, they took 300 random names from 3

[22]`https://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html`
[23]`https://www.d.umn.edu/~tpederse/namedata.html`
[24]`https://catalog.ldc.upenn.edu/LDC2014T16`

sources (US Census, Wikipedia, and computer science conference program committees) and downloaded the top 200 webpages for each name (Artiles et al., 2010). Many of these webpages do not contain much useful text (e.g., Amazon lists or University member lists) which makes them not suitable for our task.

Some researchers have created several freely available[25] pseudo-ambiguity datasets by extracting documents from the English Gigaword corpus and then conflating names (e.g., Pedersen et al. (2005); Kulkarni and Pedersen (2005); Pedersen et al. (2006)). Similar to the work by Kulkarni (2006), the data does not contain the entire documents, but only a fixed window of words around the entity. Since our approach aims at extracting context from the entire document, these datasets are not suitable for us.

In the end we decided to create our own pseudo-ambiguity dataset for several reasons:

- Identifying unambiguous entities in datasets is not a hard task. We are interested in how well our approach works on the actual ambiguous cases. A dataset that contains only ambiguous entities allows us to better study this task.
- The approach might work exceptionally well or bad on certain entities. To get a better idea of how it works in general we do not want to limit ourselves to just one entity but we want to test on several entities.
- Using a pseudo-ambiguity dataset gives us an analytical advantage: it allows us to experiment with enough data to study different corpus sizes as training material without the need to annotate a large dataset by hand.
- Manual annotation of linguistic information (e.g., coreference information) is usually not available for real applications. Thus, we use automatic annotation with NLP tools for more realistic results.
- Manual annotation for named entities is also usually not available and very time-consuming to annotate, especially for a large corpus.

Pseudo data was originally proposed for disambiguating word senses (Schütze, 1992; Gale et al., 1992) and has been used for named entity disambiguation tasks before (e.g., Mann and Yarowsky (2003); Gooi and Allan (2004)).

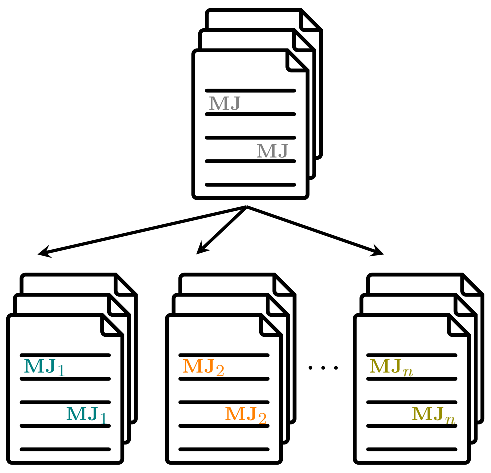
The big advantage of pseudo data compared to real data is that it is a fast and inexpensive way to obtain a very large amount of data for training and testing without having to annotate it manually. This allows us to experiment with different corpus sizes and helps us analyze the minimum and optimal numbers of training documents required to achieve reliable results. These numbers can be taken as an indicator for a good size when creating a dataset consisting of real data for the task.

[25] `https://www.d.umn.edu/~tpederse/namedata.html`

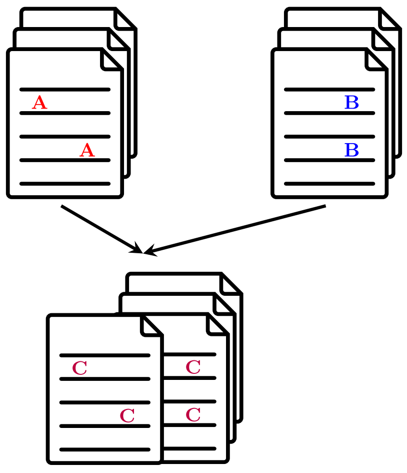
Another advantage is that it is possible to obtain a high amount of data about people who are quite similar. Very often, people with the same name are quite different (e.g., *"Michael Jackson, the American singer"* and *"Michael Jackson, the Canadian actor"*), which makes disambiguating them easier. There are of course people who are more similar, for example, there is a *"Michael Jackson, the English singer"*. But often these people are rather unknown and finding enough training data about them for a standard dataset is more difficult. With a pseudo-ambiguity dataset we can choose very similar people that have a high frequency to obtain many examples for our training and test sets.

3.3.2. Creating and using a pseudo-ambiguity dataset

This section explains the differences between a pseudo-ambiguity dataset and a standard dataset, the steps needed to create a pseudo-ambiguity dataset, and how it is used for evaluation.



(a) Standard dataset



(b) Pseudo-ambiguity dataset

Figure 3.1.: Creation process of a standard dataset and a pseudo-ambiguity dataset.

Figure 3.1a shows how a standard dataset is created for a named entity disambiguation task. The first step is to take a collection of documents with mentions of proper names, e.g., *"Michael Jackson"* (MJ). This collection can be obtained, for example, from an existing corpus or knowledgebase, or by scraping the web for documents. The next step is to annotate these proper name mentions with their corresponding real world entities (**MJ_1**, **MJ_2**, ..., **MJ_n**). The disambiguation task is then to decide which real world entity the name refers to.

Creating a pseudo-ambiguity dataset (Figure 3.1b) uses a different approach, which works as follows:

1. Decide which persons to include in the dataset and how to pair them. For our dataset we use two persons **A** and **B** for each entity pair, but it is also possible to use three or more persons.
2. Extract documents that contain information about the selected persons.
3. Replace all occurrences of the first/last names of the target entities **A** and **B** with an artificial pseudo name **C**.
4. The task is now to disambiguate these artificial pseudo names **C**.
5. The original names **A** and **B** are used as gold labels to evaluate the results.

The following examples will illustrate on real data, taken from the English Gigaword corpus, how the replacement step works.

(3.5) O_1: Israelis overwhelmingly elected **Ehud Barak**, the Labor Party leader, as prime minister on Monday. **Barak** has the option to form a government without any ultra-Orthodox representation.

O_2: **Ariel Sharon** has said he is determined to keep the city of Jerusalem in Israeli hands. **Sharon** was fired as defense minister in 1983 after a commission found him indirectly responsible for the 1982 massacre [...].

(3.6) R_1: Israelis overwhelmingly elected **EhAri BarShar**, the Labor Party leader, as prime minister on Monday. **BarShar** has the option to form a government without any ultra-Orthodox representation.

R_2: **EhAri BarShar** has said he is determined to keep the city of Jerusalem in Israeli hands. **BarShar** was fired as defense minister in 1983 after a commission found him indirectly responsible for the 1982 massacre [...].

Example 3.5 shows the original text of two text snippets taken from the English Gigaword corpus about two people, *"Ehud Barak"* (O_1) and *"Ariel Sharon"* (O_2). People sometimes are referred to with their full name, sometimes with their last or first names only. It is important to not only replace the last name but also the first name because otherwise the first name would be a quite distinct feature for disambiguating the last name. It is also important to choose artificial names that are unique so they are not confused with normal words or names. Our replacement algorithm works as follows on this example:

- Replace the two last names **Barak** and **Sharon** with the artificial name **BarShar**.
- Replace the two first names **Ehud** and **Ariel** with the artificial name **EhAri**.

Example 3.6 shows the text snippets after creating the artificial ambiguity by replacing the first and last names for *"Ehud Barak"* (R_1) and *"Ariel Sharon"* (R_2). The task is now to disambiguate the ambiguous artificial name *"EhAri BarShar"* (or *"EhAri"* or *"BarShar"* if only one part of the name is mentioned). The original names from before the replacement are used as the gold labels for the evaluation.

3.3.3. Name selection

For our pseudo-ambiguity dataset we selected pairs of persons with different names but very similar backgrounds, for example, the same nationality and the same profession. Choosing very similar people makes the disambiguation task harder and gives better insight into how well our approach works.

Person A	Person B	Nationality	Profession
Al Gore	Dick Cheney	American	Vice President
Ehud Barak	Ariel Sharon	Israeli	Prime Minister
Slobodan Milosevic	Vojislav Kostunica	Yugoslav/Serbian	President
Curtis Martin	Marshall Faulk	American	Running back
Peyton Manning	Joe Montana	American	Quarterback

Table 3.3.: Name pairs used for our pseudo-ambiguity dataset.

Table 3.3 shows which names[26] we extracted for our pairs and the properties that make them similar (nationality and profession). Some of these people had more than one profession over the years, however, we only list the profession that is predominant in the data we use.

As Gale et al. (1992) noted, using already ambiguous words or names as the basis for creating pseudo data would introduce unwanted ambiguity. For example, the word *"bat"* can mean a flying mammal of the order Chiroptera or an object made for hitting (e.g., a baseball bat), among other meanings. The contexts these words occur in, however, are usually very different. The goal of using pseuo data is to disambiguate an artifical word. If one (or both) of the original words are also ambiguous, it is harder for a classifier to learn properties about it. For this reason we made sure to choose names that do not have other famous name bearers which appear in Wikipedia. Only *"Al Gore"* has a disambiguation page in Wikipedia, however, the other two people with the name are usually referred to as *"Albert Gore Sr."* and *"Albert N. Gore"*. All other names do not

[26]The names *"Slobodan Milošević"* and *"Vojislav Koštunica"* are written as *"Slobodan Milosevic"* and *"Vojislav Kostunica"* in this thesis as this is how they are spelled in our data.

have disambiguation pages in Wikipedia. We assume that only entities mentioned in Wikipedia are prominent enough to be mentioned in the documents of our news corpus.

3.4. Experiments

In this section we present experiments done on pseudo-ambiguity data as well as a comparison to experiments done on real data. Section 3.4.1 describes the heuristics we use for extracting documents and how we create a small dataset with real data. In Section 3.4.2 we present the tools we use and describe the baseline. We also discuss a possible problem with the evaluation and how to solve it. In Section 3.4.4 we discuss the results of our experiments on both pseudo-ambiguity data and real data as well as the parameter investigation. Finally, we present an error analysis in Section 3.4.5.

3.4.1. Data and preprocessing

We used subsets of two corpora for our experiments with pseudo-ambiguity data:

- OntoNotes 4.0[27] (Weischedel et al., 2011) in the CoNLL 2011 format (Pradhan et al., 2011)
- English Gigaword[28] (Graff and Cieri, 2003) in SGML (Standard Generalized Markup Language) form

We also compared our results on pseudo data with results of the experiments we perform on real data. For this purpose, we extracted documents from the English Gigaword corpus about two American football players named *"Roy Williams"* (one of them a safety, the other one a wide receiver) and annotated the name-entity references manually.

In this section we describe the corpora we use in general, how we extract documents for our task, the subsets we extracted, and the preprocessing steps taken before the experiments.

OntoNotes – pseudo data

The OntoNotes project (Hovy et al., 2006b; Pradhan et al., 2007), a collaboration between BBN Technologies and several universities, aims at developing a large corpus consisting of different types of text in three languages (English, Chinese, and Arabic), annotated with linguistic information. OntoNotes 4.0 (Weischedel et al., 2011) is the

[27] `https://catalog.ldc.upenn.edu/LDC2011T03`
[28] `https://catalog.ldc.upenn.edu/LDC2003T05`

fourth release of the project and expands the English and Chinese data from earlier versions by adding more documents to the existing types of text (newswire, broadcast news, broadcast conversation) and adding web data (weblog and newsgroups data). For Arabic, more newswire data was added. The corpus is manually annotated with linguistic information, such as POS tags, parse trees (constituency based), predicate-argument structures, word sense information, and coreference information.

	# words			
Text type	English	Chinese	Arabic	Total
Newswire	600,000	250,000	300,000	1,150,000
Broadcast news	200,000	250,000	-	450,000
Broadcast conversation	200,000	150,000	-	250,000
Web text	300,000	150,000	-	450,000
Total	1,300,000	800,000	300,000	2,400,000

Table 3.4.: Statistics for OntoNotes 4.0.

Table 3.4 shows statistics for the OntoNotes 4.0 release. The English part is the biggest with 1,300,000 words in total, followed by Chinese with 800,000 words. The Arabic part only comprises newswire data with a total of 300,000 words. Overall, OntoNotes 4.0 consists of 2.4 million words.

We did not use OntoNotes 4.0 in the original version but in the CoNLL 2011 format (Pradhan et al., 2011). In OntoNotes, linguistic information is split into individual files (one file per layer, per document). To be consistent with previous CoNLL formats, the CoNLL 2011 format merged all information from the individual files into one file type (with the file extension `.conll`). This format provides easy row by row access to the linguistic information of each sentence in the corpus. It should be noted that a small number of documents from the original OntoNotes 4.0 dataset could not be annotated for the CoNLL 2011 dataset and has been left out. More information about what was not used for the dataset can be found in the description of the shared task (Pradhan et al., 2011).

Some documents in the OntoNotes corpus are very long and had to be split into several smaller parts for annotation purposes. Not all parts are joined after the annotation process, which results in more parts than documents. According to Pradhan et al. (2011), coreference chains are coherent in these parts, and they treat each part as a separate document. We kept this approach and treated each part as separate a document as well.

For our preliminary experiments we extracted samples of two pairs of people, *"Al Gore/Dick Cheney"* and *"Slobodan Milosevic/Vojislav Kostunica"*. We did not use the

	Gore	Cheney	Milosevic	Kostunica
Train	98	34	123	94
Test	24	9	31	24
Total	122	43	154	118

Table 3.5.: Numbers of samples extracted from OntoNotes.

train/development/test partitions provided by the CoNLL 2011 format because the distribution of the extracted entities over the partitions is not regular. Instead, we split all extracted entities into a train and a test set with the ratio 80:20. Table 3.5 shows the numbers of samples we extracted from OntoNotes for those two pairs and how they are divided into our train and test set.

English Gigaword – pseudo data

Since our preliminary experiments with OntoNotes showed that a dataset that is too small does not provide sufficient information to make a reliable statement about our approach, we used a bigger dataset for further experiments. We chose the English Gigaword corpus, a very large newswire corpus that provides plenty of documents about the persons we selected for our experiments.

Source	# tokens	# docs
AFE	170,969	656,269
APW	539,665	1,477,466
NYT	914,159	1,298,498
XIE	131,711	679,007
Total	1,756,504	4,111,240

Table 3.6.: Statistics for English Gigaword.

The English Gigaword corpus (Graff and Cieri, 2003) comprises approximately 4.1 million documents of English newswire data from four international sources: Agence France Press English Service (AFE), Associated Press Worldstream English Service (APW), The New York Times Newswire Service (NYT), and The Xinhua News Agency English Service (XIE). The corpus is not annotated with linguistic information, i.e., the documents are available as raw text only. For our experiments we annotated documents with linguistic information, including coreference, automatically to gain more realistic results as gold labels are usually not available in real applications. Table 3.6 presents statistics about the English Gigaword corpus.

To create the pseudo-ambiguity dataset, we automatically extracted documents that mention the persons we selected in Section 3.3.3. For the extraction we used some heuristics to guarantee the existence of at least two mentions in each extracted document, thereby ensuring that the coreference chain is not a singleton. For each person P (where $P \in [A, B]$) of our selection, we extracted documents d from a document collection D by using the following heuristics:

```
IF the full name of Person P occurs at least X times in document d
AND IF the last name of Person P occurs at least Y times in document d
THEN extract document d for Person P
```

If the values for X and Y are high, a document contains many relevant sentences about the selected person. We experimented with different values, and extracted different subsets of documents depending on these values. This allowed us to analyze different sizes of r-context and whether a mention can still be disambiguated if it is not the main topic of the document (which can be the case if it is only mentioned once or twice).

Docs	100	100	100	200	200	200	300	300	300
Y	2	3	4	2	3	4	2	3	4
Name	# samples								
Gore	405	589	787	1084	1320	1529	1867	2293	2311
Cheney	649	917	1214	1133	1569	2286	1600	2151	3199
Barak	696	877	915	1267	1581	1891	1790	2193	2909
Sharon	523	611	750	1056	1313	1649	1599	2000	2820
Milosevic	517	609	683	1060	1303	1329	1462	1890	2129
Kostunica	533	736	821	1203	1346	1521	1635	1978	2206
Martin	505	689	805	932	1361	1615	1448	1917	2304
Faulk	398	599	872	793	1240	1357	1314	1686	1789
Manning	447	799	869	1015	1415	1700	1528	2082	2386
Montana	550	710	867	1131	1665	1821	1879	2196	2279

Table 3.7.: Number of samples from English Gigaword used for pseudo-ambiguity experiments. All numbers are counts.

Table 3.7 shows the number of samples, i.e., the proper noun mentions that we want to identify, which we obtained by extracting different numbers of documents ("Docs") and different values for the extraction heuristic parameter ("Y"). We set the parameter $X = 1$ for all our experiments but investigated different values of Y. The numbers change depending on two the parameters. Taking more documents automatically increases the number of samples. Increasing the parameter Y also increases the samples as a higher value requires more mentions of the entity in each document.

English Gigaword – real data

For the real data experiments, we created a small dataset for two American football players that both share the same name *"Roy Williams"* but different player positions (safety and wide receiver, respectively). We first extracted documents about all persons with the name *"Roy Williams"* from the English Gigaword corpus, using the same heuristic as for the pseudo-ambiguity data (with $X = 1$ and $Y = 2$[29]). All extracted documents were manually checked to determine whether they were about one of these two persons and annotated accordingly. Documents about other persons with the same name were discarded. This resulted in a total of 52 and 36 documents, respectively.

To make it more comparable to our experiments with pseudo data (minimum of 100 documents), we added documents from *Annotated English Gigaword*[30] (Napoles et al., 2012). Annotated English Gigaword is the annotated version of the original English Gigaword Fifth Edition[31], an extension of previous versions of English Gigaword (1-4). It consists of almost 10 million documents (with over 4 billion words) from several news sources and has been annotated automatically in a very similar way to our annotation of the normal English Gigaword corpus. One difference is that we used the entire pipeline of the Stanford CoreNLP system (Manning et al., 2014) for the annotation process while Napoles et al. (2012) only used the tokenizer and coreference system from the Stanford tools. For all other preprocessing steps they used different tools. Another difference is that they skipped all sentences with more than 100 tokens. According to Ferraro et al. (2014), the sentence splitter tool Splitta (Gillick, 2009), which was used for annotating Annotated English Gigaword, had a bug which resulted in a loss of 2% of the sentences[32].

We extracted documents from Annotated English Gigaword about all persons with the name *"Roy Williams"* the same way as we extract them from English Gigaword. To match them to our annotations we extracted the raw text and annotated these texts with the Stanford tools like we did for the other documents.

[29]Increasing the value of Y results in extracting fewer documents; even when adding documents from Annotated English Gigaword it results in less than 100 documents, which makes it not comparable to the pseudo data experiments.

[30]`https://catalog.ldc.upenn.edu/ldc2012t21`

[31]`https://catalog.ldc.upenn.edu/ldc2011t07`

[32]Annotated English Gigaword was not available to us at the time when we performed our experiments. After it became available, we decided against using it because (i) for the experiments with pseudo-ambiguity data, English Gigaword is sufficient and the additional documents added in later Gigaword versions are not necessary, and (ii) the loss of sentences with more than 100 tokens means a possible loss of important sentences that give information about the entities we want to identify.

Docs	100	131/118
Y	2	2
Name	**# samples**	
Williams (S)	426	553
Williams (WR)	414	481

Table 3.8.: Number of samples from (Annotated) English Gigaword used for real data experiments. The number changes depending on two parameters: number of documents (Docs) and the extraction parameter Y. All numbers are counts.

We obtained a total of 249 documents for the two persons *"Roy Williams"*: 131 documents about the saftey (S) and 118 documents about the wide receiver (WR). Table 3.8 lists the number of samples that we extracted. When limiting the documents to 100 for each person (to match it with the number of documents used in the pseudo data experiments) we obtained 426 samples (S) and 414 samples (WR), respectively. When taking all documents, the number of samples were 533 (S) and 481 (WR), respectively.

Preprocessing

Since the OntoNotes corpus provides linguistic information (including coreference information), no special preprocessing step to annotate the corpus was necessary and we could simply extract documents that contained the entities. The CoNLL 2011 format we used combines all linguistic information in one file which allows easy row by row access for each sentence. For our experiments we extracted the coreference information for each of our candidate persons, i.e., mentions and coreference chains.

The English Gigaword corpus consists of raw text and needed to be annotated. We first went through the raw corpus and extracted documents about our target entities listed in Section 3.3.3 by using the heuristics described in Section 3.4.1, creating different subsets of data for each person, depending on the variables X and Y used in the heuristics. We then annotated the extracted documents with the Stanford CoreNLP[33] system (Manning et al., 2014), version 1.3.4, with the following tools:

- Tokenizer
- Sentence Splitter
- Part-of-Speech Tagger (Toutanova et al., 2003)
- Named Entity Recognition (Finkel et al., 2005)
- Coreference Resolution (Raghunathan et al., 2010; Lee et al., 2011; Recasens et al., 2013; Lee et al., 2013)

[33]`https://stanfordnlp.github.io/CoreNLP/`

The Stanford Deterministic Coreference Resolution System uses several sieves of features, aiming at achieving high precision results for coreference. We use all sieves including pronoun matches to annotate our data with coreference information. More information on the sieve system can be found in Section 2.3.2.

3.4.2. Experimental setup

Classification with WEKA

Our goal is to determine the real world entities that the ambiguous proper names in our datasets refer to. For this we used different classification methods. The gold labels for the pseudo-ambiguity data are provided by the original names before the replacement with the pseudo name (cf. Section 3.3.2 for more information). The gold labels for the real data were annotated manually.

We used the machine learning software WEKA[34] (Hall et al., 2009), version 3.6.12, and the following classifiers which the software provides:

- Naive Bayes (John and Langley, 1995)
- J48 (Decision Tree C4.5 implementation (Quinlan, 1993))
- SMO (John Platt's sequential minimal optimization algorithm for training a support vector machine with some modifications (Platt, 1998; Hastie and Tibshirani, 1998; Keerthi et al., 2001))

To classify the data we had to convert it into the WEKA file format, called ARFF (Attribute-Relation File Format). ARFF describes a list of instances (in our case proper names) sharing a set of attributes (in our case the context the proper name occurs in as a string). While nominal attributes can be processed directly in WEKA, this does not apply to string attributes because they contain arbitrary textual values. To process string attributes, a WEKA filter needs to be applied on the data first. We used the `StringToWordVector` filter that converts string attributes into a set of attributes representing word occurrence information from the text contained in the strings. The filter's setting includes stopword removal. The default stopword list is based on Rainbow[35], a program for statistical text classification. It contains 524 common words, such as *"the"*, *"a"*, *"and"*, and *"of"*.

[34] `https://www.cs.waikato.ac.nz/ml/weka/`
[35] `https://www.cs.cmu.edu/~mccallum/bow/rainbow/`

Baselines

We used two fixed context windows as baselines (BL):

- **00 BL:** The first baseline uses the sentence that contains the entity with no further context (= base context).
- **11 BL:** The second baseline expands the 00 baseline. It adds the previous sentence and next sentence (if available) as additional context to the 00 baseline sentence.

The idea behind the second baseline is that sentences that are close to the mentioned proper name are more likely to contain additional relevant information. This corresponds to approaches that take a fixed window around the entity and gives us a stronger baseline than taking no further context. We decided to take an entire sentence instead of n words because our method is sentence-based and a word window might cut off important information at the beginning (respectively at the end) of a sentence.

In addition, we used a document-based baseline for experiments that compare our r-context against the information of the entire document:

- **Docs BL:** This baseline uses the information from the entire document.

3.4.3. Evaluation

To evaluate the preliminary experiments with OntoNotes, we split the extracted sentences into a training and test set with the ratio 80:20. For the experiments with English Gigaword, we evaluated using k-fold cross-validation.

In k-fold cross-validation, a dataset $\mathbf{D}$ is divided into k disjunct parts (also called folds) $\mathbf{D}_1$, $\mathbf{D}_2$, ..., $\mathbf{D}_k$ of equal size. The distribution of data into the folds is done randomly. The experiment is then performed k times on the data, each time using a different fold $\mathbf{D}_t$ as test set while training on the remaining folds $\mathbf{D}\backslash\mathbf{D}_t$.

In our case it was not possible to split the data into k folds randomly because we observed the “twinning” phenomenon in our data. Twinning means there are duplicate samples (or nearly duplicate samples) in the data. This happens in our dataset because of how we build the context for a proper name. Consider the following example:

(3.7) S$_1$: X_{NNP}
S$_2$: X_{PRP}
S$_3$: —
S$_4$: X_{NNP}
S$_5$: X_{NN}

Example 3.7 is a document consisting of five sentences S_1 to S_5. The proper name X appears in four of the sentences. The POS tag of each mention is noted in subscript. In our task we disambiguate all proper names X_{NNP}, i.e., in this example the mentions in S_1 and S_4. We build the context for the baselines and the filters as listed in Table 3.9.

Baseline	Context $X_{\text{NNP}}(S_1)$	Context $X_{\text{NNP}}(S_4)$
00	S_1	S_4
11	S_1S_2	$S_3S_4S_5$
Docs	$S_1S_2S_3S_4S_5$	$S_1S_2S_3S_4S_5$
r-context filter	Context $X_{\text{NNP}}(S_1)$	Context $X_{\text{NNP}}(S_4)$
NNP	S_1S_4	S_1S_4
NNP+NN	$S_1S_4S_5$	$S_1S_4S_5$
NNP+PRP	$S_1S_2S_5$	$S_1S_2S_5$
NNP+NN+PRP	$S_1S_2S_4S_5$	$S_1S_2S_4S_5$

Table 3.9.: Twinning phenomenon in the data when creating contextes.

Table 3.9 shows the context for the proper name in sentence S_1 in the second column and the context for the proper name in sentence S_4 in the third column. The context for the window-based baselines is different because the base sentences as well as the previous and next sentences are different. For the document-based baseline both contexts are the same because in both cases all sentences from the document are taken. The r-context we create with our filter methods is the same in both cases as well. This happens because both entities are coreferent, i.e., the coreference chain and with this the sentences we extract for the context are the same for both entities.

Twinning becomes problematic when the duplicate samples occur in both the training and the test set. If a classifier sees a sample during training, it can easily decide on the same sample during the test phase. This can lead to overestimation of validity of the model, a bias on the results, and possible overfitting of the classifier as has been mentioned in previous work (e.g., Forman and Scholz (2010); Brenning (2012); Karimi et al. (2015)).

To resolve the twinning problem, we do not create the k folds in a completely random way but with an algorithm that ensures no duplicates are distributed over several folds. To ensure this, the algorithm treats all contexts created from the entities in a coreference chain as a cluster and places them into one fold. For our experiments we choose $k = 5$.

The first version of this algorithm, which we also used for our previous publication of this approach (Glaser and Kuhn, 2014), first sorted all context clusters by the number of contexts. Then, it iterated over the k folds and placed one context in each fold, repeating this iterating procedure until the maximum size of each folds was reached. By doing this, the context clusters were distributed over the folds more equally, avoiding cases where some folds have many duplicate contexts while others have less duplicates.

We changed the way this algorithm works later. Instead of distributing the context clusters equally sorted by cluster size in descending oder, the algorithm now distributes them randomly into the folds until each fold reaches the maximum size. This means that it is possible that some folds end up with many big context clusters while others could have many small ones. This change stems from our belief that this random distribution is closer to the standard way of creating folds for cross-validation.

For all results we report the weighted micro-averaged F_1 score over both entities in each entity pair experiment for both pseudo-ambiguity and real data experiments. To measure statistical significance[36], we use the 5x2cv paired t test (Dietterich, 1998). We compare the results of our system (each result for every filter method) against the stronger baseline (11 baseline) and measure whether the difference of the scores is statistically significant at $p < .05$.

3.4.4. Results and discussion

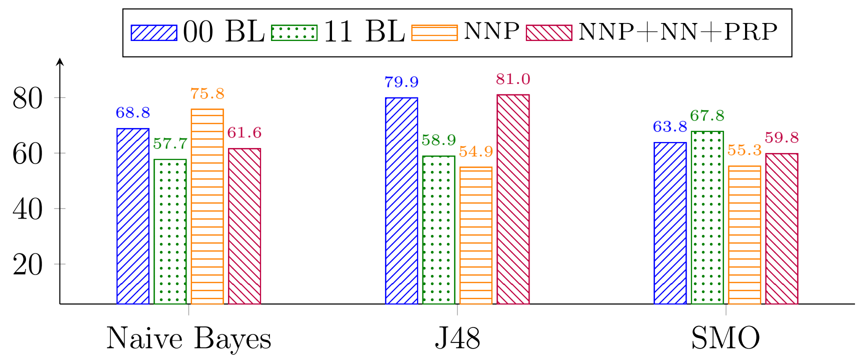
In this section we discuss the results of our experiments. We first show the results for the preliminary experiments conducted on pseudo-ambiguity data extracted from OntoNotes. Then, we present the results for the pseudo-ambiguity data experiments with the data from the English Gigaword corpus. After discussing general results, we investigate the usefulness of different parameters (number of documents, different extraction methods). Finally, we show results for experiments done with real data from the English Gigaword corpus and compare it to results on pseudo-ambiguity data.

Experiments on pseudo data using OntoNotes

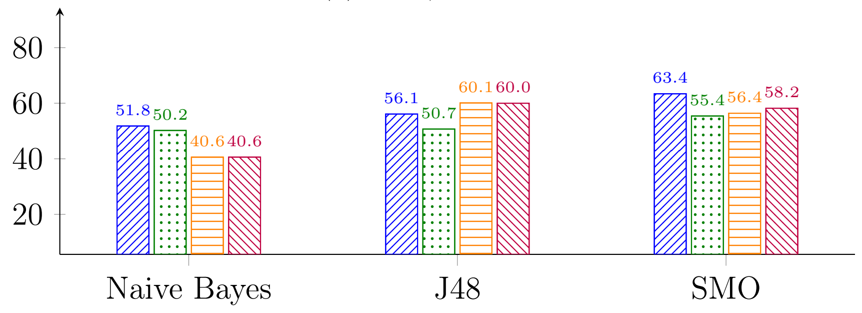
We conducted preliminary experiments on pseudo-ambiguity data with two pairs of people (*"Gore/Cheney"* and *"Milosevic/Kostunica"*), extracted from the OntoNotes corpus. In these experiments we tested two of the four final filters: NNP (sentences with the entity as proper noun only) and NNP+NN+PRP (all sentences with the entity). Figure 3.2 shows results for our preliminary experiments on OntoNotes. For both pairs of entities the results of the baselines and our system vary greatly and there is no clear trend of which one is better. In some cases, our system produces the better results, in other cases the baselines are better.

Looking at these results, the first impression is that r-context is not better than a simple fixed context window around the entity and in some cases even worsens the results.

[36]Statistical significance of the results is given in the detailed results in Appendix A.



(a) Gore/Cheney



(b) Milosevic/Kostunica

Figure 3.2.: Results of preliminary experiments on pseudo-ambiguity data (OntoNotes).

However, we attribute this to the fact that the sentences extracted from OntoNotes produced a very small dataset. As listed in Table 3.5, the total number of obtained samples for each person ranges from 43 to 154. The number of samples in the test set ranges from 9 to 31. We pointed out in Section 3.4.3 that we had to be careful with the twinning problem in our data, and had to divide samples in a way that they do not appear in both the train and test set. This resulted in all duplicate samples being in one split of the dataset. If one of these duplicate samples in the test set is misclassified, all of its duplicates will be misclassified as well. This hurts the classification results and has an especially big impact on a small test set.

The baselines are less affected by the twinning problem. The 00 baseline generally has unique contexts. The 11 baseline can have sentence overlap in some cases (e.g., if two entities are in subsequent sentences, then the 11 baseline for both entities shares these sentences). Since the baselines are usually not exactly the same (an exception would be in very small documents with only 2-3 sentences), misclassifying one sample in the test phase does not necessarily mean a similar sample will be misclassified as well. For these reasons, the baselines can achieve better results in some cases.

Our preliminary tests show the limitations of a small dataset, especially if twinning is involved. The number of obtained samples (43-154 for each person) is too small to make reliable statements about the usefulness of our r-context. In the next part of this section we discuss experiments we perform on a much bigger dataset, and show that our approach of using r-context does work well.

Experiments on pseudo data using English Gigaword (document baseline)

In our first experiments using the English Gigaword corpus, we used the entire document as the baseline. Due to the twinning problem that we described in Section 3.4.3, the experimental construction is slightly different compared to the experiments that use a fixed window as the baseline. In the fixed window baseline experiments, we evaluate each mention of the person (as a proper name) using our r-context against the two fixed window baselines. In a window-based baseline approach, the baseline context for each mention is different, while it is always the same when taking the entire document as the context. To avoid duplicate contexts in our training and test data we constructed the experiments in a way that for each document we only disambiguate the first mention of the person.

Table 3.10 shows the results for the entity pair *"Gore/Cheney"*, using all three classifiers (Naive Bayes (NB), J48, SMO) with different parameters for documents (100, 200, 300) and extraction ($Y \in \{2, 3, 4\}$). In almost all cases taking our r-context gives better results than taking the entire document as the context. One might expect that taking the entire document provides better results than taking only parts of the document. However, the entire document does not only contain useful information for disambiguating the entity in question, but often also adds a lot of noise. Our results in Table 3.10 confirm that taking the entire document does not always yield the best results and that taking a more selective context such as our r-context is often the better choice.

There are two exceptions in our results in Table 3.10, where taking the entire document performs better (300 docs, $Y = 4$, NB or SMO as classifiers). In a few cases some types of r-context yield better results than the baseline, while other types perform equally or worse. In general, this happens when the extraction parameter Y is high. Documents that were extracted by using a high extraction parameter contain many mentions of the entity in question and are therefore more likely to be concerned mostly about the entity. If a document is mostly about one entity, then taking the entire context can add more information. A smaller extraction parameter means that many of the extracted documents mention the entity fewer times, which can indicate that the document is mostly about other entities. In this situation, taking our r-context is more useful to extract precisely the parts from the document that are concerned with the entity in question.

			Baseline	Filter Methods			
Class.	Y	Docs	Docs	NNP	NNP+NN	NNP+PRP	NNP+NN+PRP
NB	2	100	79.4	92.5	**93.0***	90.0	88.4*
		200	80.1	91.2	91.0	**93.0**	92.7
		300	73.9	86.5	86.7	**87.1**	87.0
	3	100	84.4	**89.0**	**89.0**	84.9	84.9*
		200	80.1	**93.2**	92.7	91.5*	91.2*
		300	77.5	**89.8**	88.5	89.5*	89.3
	4	100	92.5	**96.5**	96.0*	94.0	94.0*
		200	93.0	**95.5**	94.7*	92.0	92.0
		300	**96.7**	95.7	95.5	94.0	93.5*
J48	2	100	74.0	89.5	89.5	**90.4**	**90.4**
		200	74.3	86.7	86.5	**87.7**	**87.7**
		300	76.3	81.7	82.5	**82.9**	82.8
	3	100	88.0	**90.5***	90.0*	87.9	87.9
		200	82.2	**92.2**	91.2	91.2	91.5
		300	76.2	83.6	**84.0**	82.3	83.0*
	4	100	87.9	**95.0***	**95.0***	87.9*	88.9*
		200	89.2	**95.2***	**95.2***	94.5*	94.5*
		300	91.3	**95.0***	94.0	94.5	94.0
SMO	2	100	89.9	**95.5***	**95.5***	95.0*	95.0*
		200	86.5	**93.5**	93.0	93.4	93.2
		300	78.0	88.7	88.5	88.2*	**89.0***
	3	100	90.9	**94.5**	**94.5**	94.0	93.5
		200	89.0	**95.0***	**95.0***	94.7	94.5*
		300	84.9	**91.8**	91.5	91.6*	91.5*
	4	100	97.0	98.0	**98.5**	97.0	97.0
		200	97.0	**99.0**	98.7	**99.0***	98.5
		300	**98.4**	97.5	97.5*	98.0*	98.2

Table 3.10.: Results of the experiments with r-context vs. the entire document, for the entity pair *"Gore/Cheney"*, using three different classifiers (Naive Bayes (NB), J48, SMO) with different parameters for documents (100, 200, 300) and extraction ($Y \in \{2, 3, 4\}$). All results report the weighted micro-averaged F_1 score over both entities in each entity pair. The best result in each row (i.e., parameter setting) is bold. Results of the filter methods are marked with * if the difference to the docs baseline is statistically significant at $p < .05$.

Experiments on pseudo data using English Gigaword (window baseline)

In this section we the discuss results of our experiments on pseudo-ambiguity data taken from the English Gigaword corpus and compare our approach against baselines consisting of fixed windows. We conducted several experiments with different parameters:

- 5 different pairs of people (see Section, 3.3.3, Table 3.3)
- 3 different classifiers (Naive Bayes, J48, SMO)
- 3 different numbers of documents (100, 200, 300)
- 3 different extraction values used for the heuristics ($Y \in \{2, 3, 4\}$)

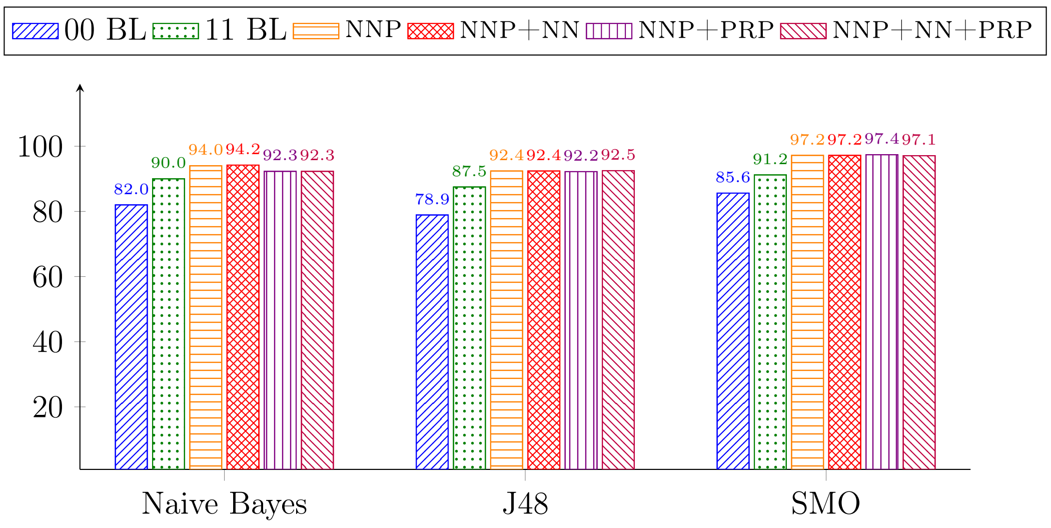
Due to the large amount of experiments, we limit the discussion to one setting of parameters to describe the general trend of the results. We chose a setting which has an average performance in the experiments to give an estimate of how the approach works in general. The results are based on experiments done after changing the algorithm that distributes samples into folds (randomly instead of equal distribution, see Section 3.4.3 for details). Consequently, the results are slightly different than in our previous publication (Glaser and Kuhn, 2014). We elaborate on the differences in more detail in Appendix A, where we also show detailed results of all experiments.

Figure 3.3 shows results of the two baselines and our system for all four types of r-context (NNP, NNP+NN, NNP+PRP, NNP+NN+PRP) for the three pairs of politicians (*"Gore/Cheney"*, *"Barak/Sharon"*, and *"Milosevic/Kostunica"*), using all three classifiers (Naive Bayes (NB), J48, SMO). The parameters in this setting are 200 documents and $Y = 4$ as the extraction heuristics.

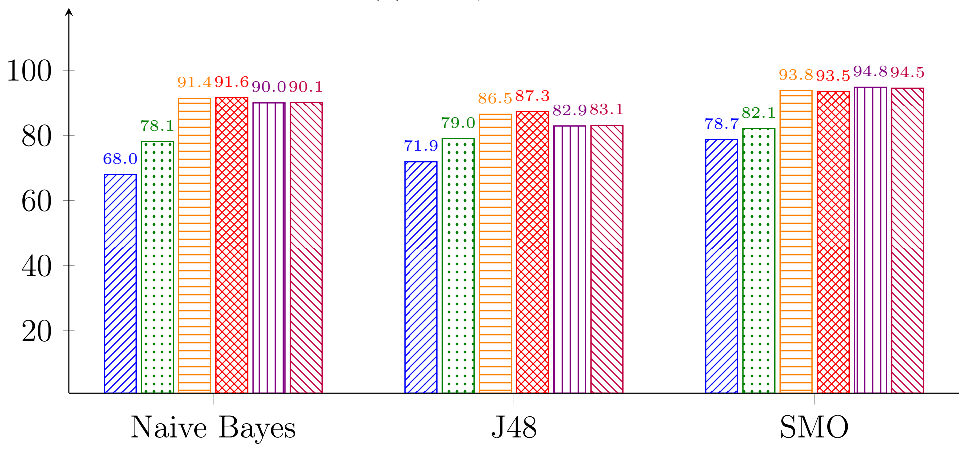
The 00 baseline ranges from 68.0% (*"Barak/Sharon"*, NB) to 85.6% (*"Gore/Cheney"*, SMO). The sentences which contain the entity provide useful information in many cases, for example, when the sentence introduces the entity and gives more background related to them. This simple approach still fails in more than 14.4% to 32.0% of the cases because the information provided in the sentence is not sufficient. This happens especially when a sentence simply mentions an entity while giving descriptive context about other entities.

The 11 baseline, which adds the previous and next sentences to the context, shows improvement over the simple 00 baseline. In our results in Figure 3.3 it ranges from 78.1% (*"Barak/Sharon"*, NB) to 91.7% (*"Milosevic/Kostunica"*, SMO). Overall, the 11 baseline shows better results for all our experiments compared to the simple 00 baseline, as can be seen in Tables A.1–A.5 in Appendix A. This is in line with our expectations and affirms the first hypothesis we postulated:

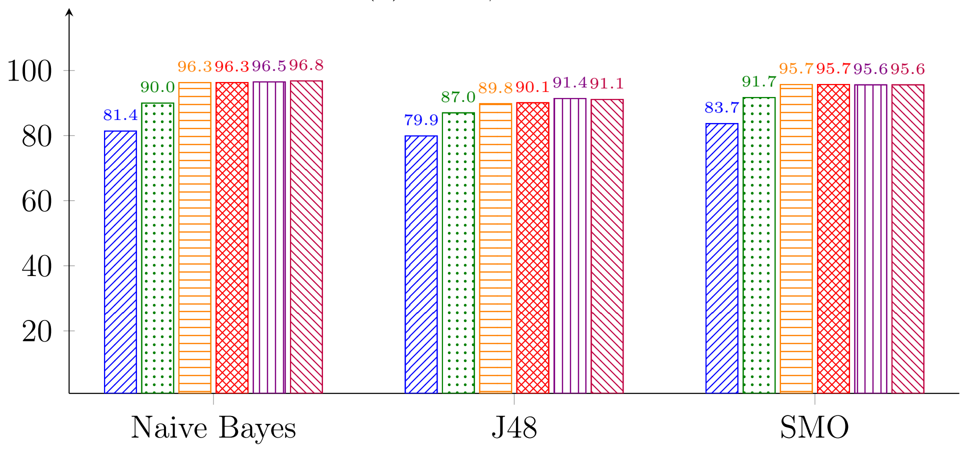
Hypothesis 1: A larger window around mentions of names provides more reliable feature information. ✓



(a) Gore/Cheney

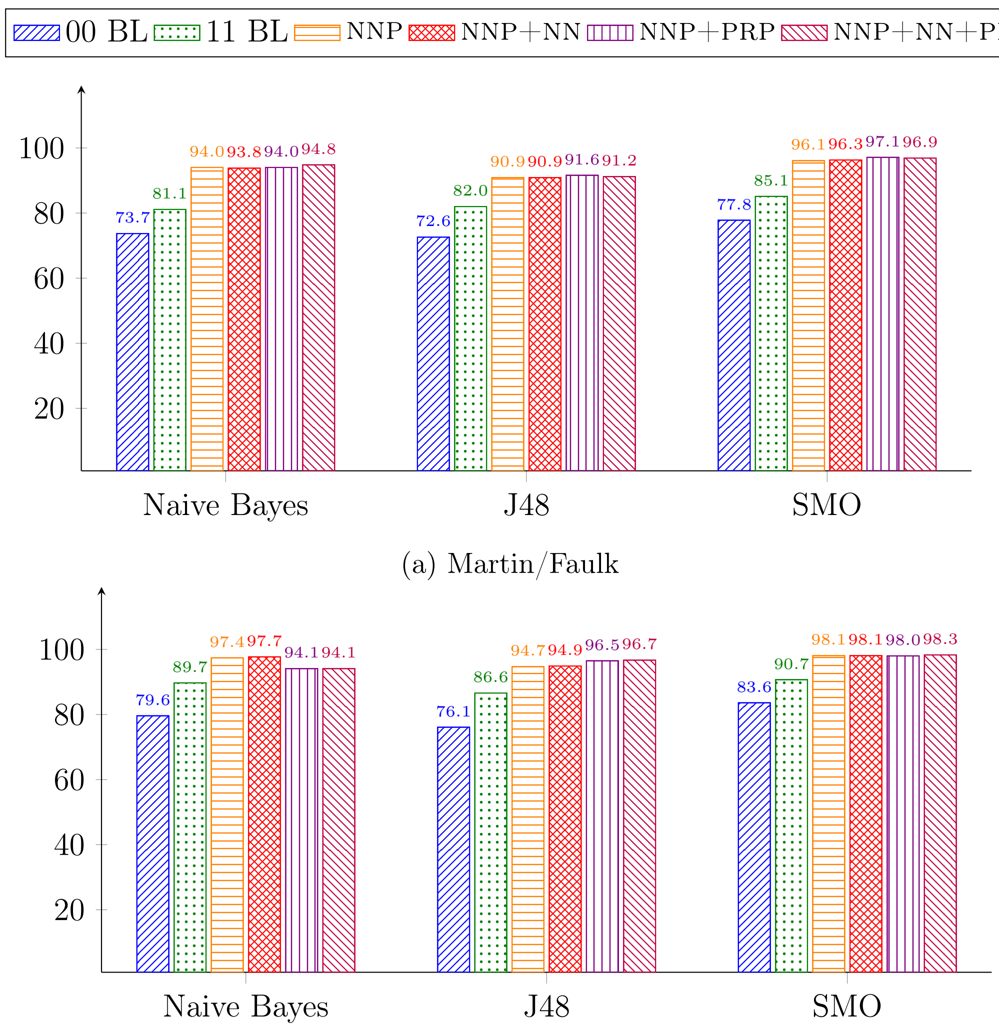


(b) Barak/Sharon



(c) Milosevic/Kostunica

Figure 3.3.: Results of experiments on pseudo-ambiguity data, for the three pairs of politicians, using three different classifiers (Naive Bayes (NB), J48, SMO), 200 documents extracted with the values $X = 1$ and $Y = 4$.



(a) Martin/Faulk

(b) Manning/Montana

Figure 3.4.: Results of experiments on pseudo-ambiguity data, for the two pairs of sportsmen, using three different classifiers (Naive Bayes (NB), J48, SMO), 200 documents extracted with the values $X = 1$ and $Y = 4$.

Besides the baselines, Figure 3.3 shows the results of our system for all four types of r-context (NNP, NNP+NN, NNP+PRP, NNP+NN+PRP). In all three subfigures, our approach that uses r-context beats both baselines in all cases. The amount of improvement depends on the entity pair and the classifier.

Figure 3.4 shows results of the two pairs of sportsmen (*"Martin/Faulk"* and *"Manning/Montana"*), using the same classifiers and settings as used above with the politicians. The pattern of results is comparable: the 11 baseline performs better than the 00 baseline and our system performs better than both baselines in all cases. This shows that our approach is not limited to a specific type of persons. It can be applied to all persons without any special training and is therefore domain independent.

In few cases the results of our system are worse than the 11 baseline (cf. the detailed results in Tables A.1, A.2, A.4, and A.5). This happens in some of the cases where (i) the extraction parameter Y is low, i.e., less samples are extracted from the documents in general, and/or (ii) fewer documents are taken. These results are in line with our preliminary experiments on the OntoNotes corpus: using a small dataset is not enough to produce reliable results. Adding more documents as well as increasing the extraction parameter (both of which gives more samples for the classification) does improve the overall results and makes our system perform better than the 11 baseline.

The extraction parameter Y also influences the size and quality of the r-context we create. The parameter defines the *minimum* number of occurrences of the last name of a person in a document for it to be extracted for the dataset. Since it is a minimum number and not a definite number, documents can also contain more occurrences of the last name. However, a higher value for Y ensures documents have a higher amount of occurrences. The full name or last name of a person can be easily identified by the coreference system via string matching. This means that the chance of a wrong coreference decision is very low and the sentence added to the r-context is actually about the entity.

Assuming that we have enough samples in our dataset to properly train a classifier, we can confirm the second hypothesis that we postulated:

Hypothesis 2: Structurally informed choice of the context window helps improving the disambiguation. ✓

One reason for the improved results is that our r-contexts consist of more sentences that mention the entity in the document, which is usually larger than the context of the baselines. The second and more important reason is the contexts are likely to contain more relevant information about the entity, while the additional sentences in the 11 baseline (or any larger fixed window around the entity) might contain only little or no relevant information.

Our results do not show a clear pattern of which type of r-context is the most helpful one, as can be seen in Figures 3.3 and 3.4. For a better overview of the irregularity of the results we refer to Tables A.1–A.5 in Appendix A.

In some cases the simplest filter method (NNP) yields the best results and extending the r-context with more sentences gained by common nouns and/or pronouns worsens the results. Correctly determining coreference of common nouns and pronouns is harder than determining coreference of proper nouns (which can be done by a simple string matching feature). If a common noun or pronoun gets incorrectly classified as coreferent, our system adds sentences to the r-context which do not provide helpful information (or in the worst case, specific information about the incorrect entity). Adding more noise and possibly clues about another entity hurts the classification performance. Since we do not have a gold annotated dataset but determine coreference automatically, this happened a few times. We discuss some cases in our error analysis in Section 3.4.5.

In other cases, one of the filter methods that extends the simple NNP context by one other POS tag (i.e., the filter methods NNP+NN and NNP+PRP) shows improvement over the NNP filter method. In these cases, the sentences added by either coreferent common nouns or coreferent pronouns improve the results of the NNP filter method, while the other one (i) does not change the results[37], (ii) improves the results slightly but not as much as the other one, or (iii) worsens the results (because of the same problem as described in the last paragraph).

The filter method NNP+NN+PRP provides the largest context as it contains all sentences with mentions of the coreference chain. However, it becomes clear from the results that the largest context is not automatically the best context. The problem is again that the additional sentences might contain too much noise or not enough useful information. This means we can confirm our third hypothesis with reservations:

Hypothesis 3: Identifying coreference chains of entities provides relevant context. (✓)

The results show that we can create relevant context by exploiting coreference chains and that using this r-context improves the results in almost all cases. There are only very few cases where the r-context performs worse than the 11 baseline and no cases where it performs worse than the 00 baseline.

Parameter investigation for pseudo-ambiguity data using English Gigaword

We conducted several experiments using different parameters (number of documents used and different values for the heuristics to extract documents) to investigate how different corpus sizes and different extraction methods influence the results. Studying these parameters is useful because (i) it can show if a very large corpus is really needed for a certain task or if a smaller corpus is sufficient and (ii) it gives more insight into the relevant contexts we create. The findings of this study can then be used for designing copora for this task in the future.

[37]This can happen (i) if the extended filter method does not add any sentences because there are no common nouns, respectively, pronouns in the coreference chain, i.e., both contexts are exactly the same, (ii) if the contexts are different but the classifier yields the same results, e.g., because the contexts are very similar, or (iii) if the results are slightly different but due to rounding up or down to one digit the numbers are the same.

We present the analysis of this investigative study as follows. First, we present results for a fixed value of the extraction parameter Y but with different numbers of documents (100, 200, and 300 documents). Second, we use a fixed number of documents and change the extraction parameter ($Y \in \{2, 3, 4\}$). Increasing either value adds more samples to our dataset. While increasing the number of documents adds more samples to the dataset than increasing Y, a higher value of Y additionally increases the chance of higher quality sentences to be added to our r-context (see the discussion in the previous subsection). For a comparison of the number of samples for each name pair and parameter setting we refer to Table 3.7 in Section 3.4.1.

Gore/Cheney		Baselines		Filter Methods			
Classifier	Docs	00	11	NNP	NNP+NN	NNP+PRP	NNP+NN+PRP
	100	81.6	**90.0**	93.0	93.0	**93.7**	**93.7**
NB	200	**82.0**	**90.0**	**94.0**	**94.2**	92.3	92.3
	300	80.3	89.7	93.8	93.4	92.4	92.3
	100	76.0	86.7	94.3	94.3	87.7	**93.4**
J48	200	78.9	87.5	92.4	92.4	92.2	92.5
	300	**79.2**	**88.4**	**95.7**	**95.9**	**93.2**	92.6
	100	84.7	90.6	98.8	98.8	98.1	98.1
SMO	200	85.6	91.2	97.2	97.2	97.4	97.1
	300	**86.7**	**91.8**	**99.1**	**99.4**	**98.8**	**99.1**
Barak/Sharon		Baselines		Filter Methods			
Classifier	Docs	00	11	NNP	NNP+NN	NNP+PRP	NNP+NN+PRP
	100	**77.1**	**86.4**	**96.2**	**96.3**	**96.8**	**96.2**
NB	200	68.0	78.1	91.4	91.6	90.0	90.1
	300	67.9	77.1	88.8	89.2	88.5	88.6
	100	**72.8**	**80.6**	**86.5**	**88.2**	**88.5**	**88.3**
J48	200	71.9	79.0	**86.5**	87.3	82.9	83.1
	300	71.5	80.2	82.3	83.1	80.3	79.1
	100	**80.3**	**86.8**	93.2	**93.6**	94.1	94.1
SMO	200	78.7	82.1	**93.8**	93.5	**94.8**	**94.5**
	300	77.9	82.5	93.7	93.0	93.3	93.5

Table 3.11.: Comparison of results of different numbers of documents used (100, 200, or 300), for the pairs Gore/Cheney (first table) and Barak/Sharon (second table), with parameters $X = 1$, $Y = 4$. The best result in each classifier group is marked in bold.

We illustrate how results vary when using different numbers of documents (100, 200, 300) in Table 3.11. The extraction parameter is set at $Y = 4$. We limit the results to two entity pairs: *"Gore/Cheney"* (first table at the top) and *"Barak/Sharon"* (second table at the bottom). Bold numbers show the best results for each classifier. For a comparison of other pairs we refer to the detailed results in Appendix A.

Taking more documents does not always help improve the results. Naive Bayes performs better on smaller datasets for both entity pairs. J48 and SMO benefit from more data for the entity pair *"Gore/Cheney"* but not for the entity pair *"Barak/Sharon"*. Comparing it with other entity pairs and other settings (see detailed results in Tables A.1–A.5), it becomes apparent that increasing the number of documents does not automatically improve the results.

Gore/Cheney		Baselines		Filter Methods			
Classifier	Y	00	11	NNP	NNP+NN	NNP+PRP	NNP+NN+PRP
	2	72.8	80.0	79.9	79.9	78.3	80.5
NB	3	76.0	84.4	85.3	84.7	85.0	84.8
	4	**82.0**	**90.0**	**94.0**	**94.2**	**92.3**	**92.3**
	2	66.0	78.8	85.4	87.6	84.7	84.7
J48	3	71.9	82.1	87.7	89.1	88.9	89.1
	4	**78.9**	**87.5**	**92.4**	**92.4**	**92.2**	**92.5**
	2	73.5	81.5	94.4	94.4	95.2	95.4
SMO	3	76.8	84.5	95.7	95.4	96.2	96.4
	4	**85.6**	**91.2**	**97.2**	**97.2**	**97.4**	**97.1**
Barak/Sharon		Baselines		Filter Methods			
Classifier	Y	00	11	NNP	NNP+NN	NNP+PRP	NNP+NN+PRP
	2	**68.8**	76.3	80.4	79.4	78.7	78.7
NB	3	68.5	77.1	86.4	87.0	86.9	86.6
	4	68.0	**78.1**	**91.4**	**91.6**	**90.0**	**90.1**
	2	65.9	71.5	80.2	75.1	80.3	75.8
J48	3	68.7	77.0	76.7	78.4	76.9	76.6
	4	**71.9**	**79.0**	**86.5**	**87.3**	**82.9**	**83.1**
	2	69.9	75.4	85.3	85.7	86.0	85.6
SMO	3	72.3	78.4	87.3	86.6	90.1	89.9
	4	**78.7**	**82.1**	**93.8**	**93.5**	**94.8**	**94.5**

Table 3.12.: Comparison of results of different extraction parameters ($Y \in \{2,3,4\}$), for the pair Barak/Sharon, with parameters 200 docs, $X = 1$. The best result in each classifier group is marked in bold.

Table 3.12 shows how the results change with different extraction parameter Y ($Y \in \{2,3,4\}$). The number of documents is set to 200 documents. The table at the top shows again results for the entity pair *"Gore/Cheney"* and the table at the bottom shows results for *"Barak/Sharon"*. The best results for each classifier are in bold. For a comparison of other pairs we refer to the detailed results in Appendix A. Increasing the extraction parameter Y systematically increases the results in almost all cases. There are two reasons for this: First, increasing Y increases the numbers of samples used in the test set in general (see Table 3.7 in Section 3.4.1). Every sentence that contains the entity mentioned by its full name or last name is added as a sample to the dataset. A higher value for Y means that documents have to contain more occurrences of the last name (compared to lower Y values), which results in more samples being added.

Second, since a higher value for Y means the entity is referred to more often with their last name in a document, the coreference chains for this entity contain more proper nouns. Proper nouns can be easily identified as coreferent, therefore the chance of misclassifications is low. Sentences gained for the r-context through correct classification of coreference are more likely to contain useful information and thus increase the overall quality of the r-context. This is why especially our system benefits from increasing Y to obtain better r-contexts.

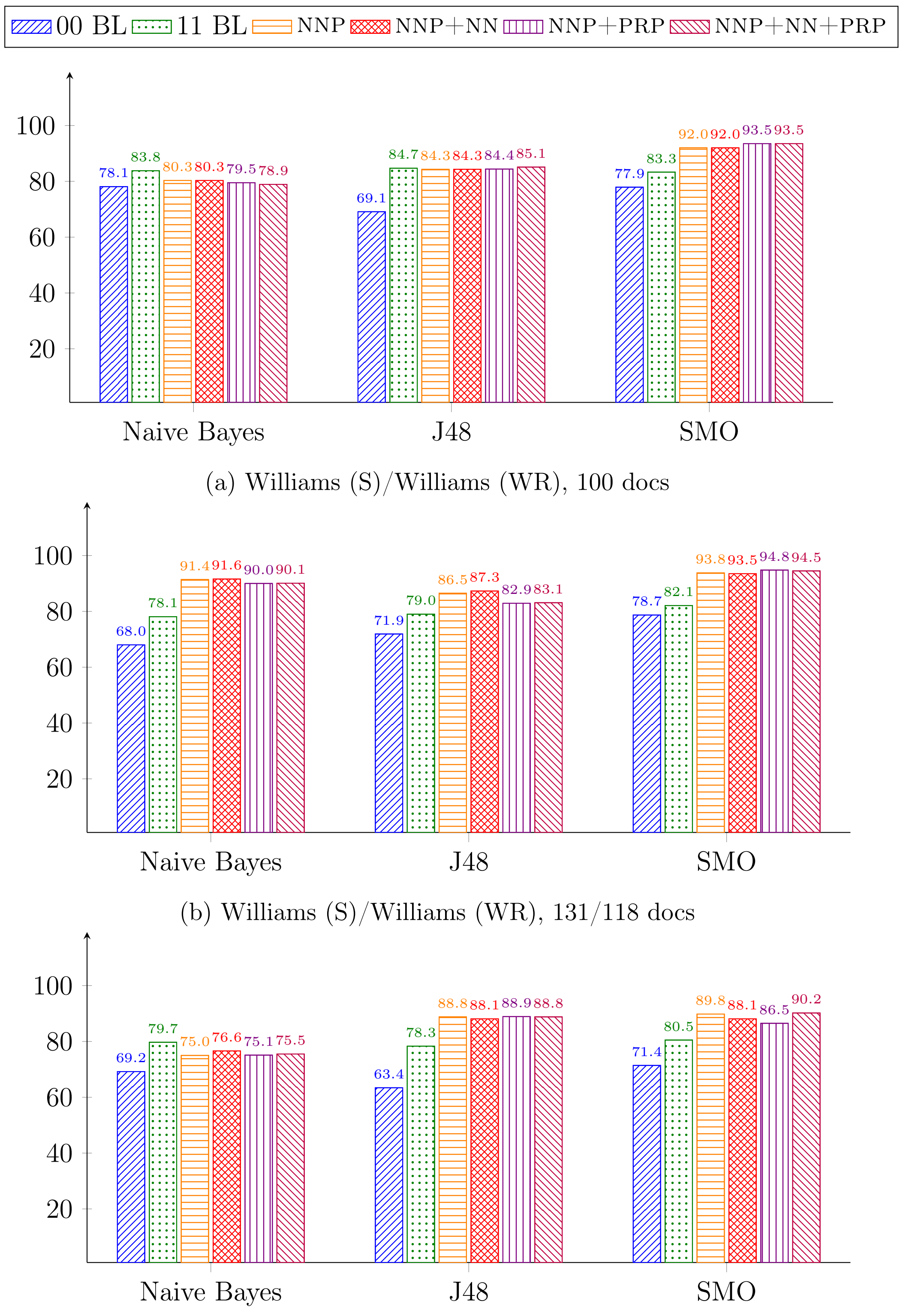
The pattern seen in Table 3.12 is quite regular and can also be observed with other entity pairs and other parameter settings (cf. Tables A.1–A.5).

Experiments on real data using English Gigaword

Results from experiments done with peudo-ambiguity data can give us valuable insight into the effects and efficiency of a new approach when not enough real data is available. However, since pseudo-ambiguity data is created artificially it might not give accurate results. To demonstrate that pseudo-ambiguity data can be used for our task, we show comparative results from a small dataset using real data (manually annotated for named entities, automatically annotated for all other linguistic information including coreference) and compare them to the results that used pseudo-ambiguity data.

Figure 3.5 shows results of our experiments on real data, for the pair *"Roy Williams (safety)/Roy Williams (wide receiver)"*, taking 100 documents (Figure 3.5a) and 131/118 documents (Figure 3.5b), as well as a comparison to results on pseudo-ambiguity data, for the pair *"Manning/Montana"* (Figure 3.5c), taking 100 documents. For a better comparison we chose a pseudo data pair that has very similar properties to the real data pair. All of them are American football players, but the the persons in the pseudo data pair have different player positions than the persons in the real data pair. As with the experiments before, we use all three classifiers.

The results on real data are comparable to results on pseudo data. The 11 baseline is generally better than the 00 baseline. Our system that uses r-context performs better



(a) Williams (S)/Williams (WR), 100 docs

(b) Williams (S)/Williams (WR), 131/118 docs

(c) Martin/Faulk, 100 docs

Figure 3.5.: Results of experiments on real data (a) & (b) and comparison to results for experiments on pseudo-ambiguity data (c), using three different classifiers (Naive Bayes (NB), J48, SMO), documents extracted with the values $X = 1$ and $Y = 2$.

than the baselines in most cases, with a few exceptions where some of the results drop below the 11 baseline (but never below the 00 baseline). This is the same behavior we observed with the results on pseudo-ambiguity data in cases where the numbers of documents and the extraction parameter Y are low (as discussed in the previous subsection about results on pseudo-ambiguity data, see also results in Figure 3.5c).

Taking the maximum number of documents (131/118) improves some of the results (when using J48) but worses the results in other cases (when using Naive Bayes or SMO). Increasing the number of documents also increases the cases where the system results are better than the 11 baseline. Since the amount of additional documents is not high, the improvements are also not high. Disambiguating these two persons (*"Roy Williams"*) is very hard because they share very similar properties besides the same name. Table 3.13 lists some of these properties.

	Roy Williams #1	Roy Williams #2
Position	Safety	Wide Receiver
Birthday	August 14, 1980	December 20, 1981
Teams	Dallas Cowboys (2002–2009)	Detroit Lions (2004–2008)
	Cincinnati Bengals (2009–2010)	Dallas Cowboys (2008–2010)
		Chicago Bears (2011)

Table 3.13.: Properties for the real data pair *"Roy Williams/Roy Williams"*.

Both players are very close in age. They also both played on the same team (Dallas Cowboys) for about six months. The team name is a very prominent feature, which makes it hard for a classifier to disambiguate samples which mention the team name they both have in common. Despite the strong similarities of these two people, we obtain good overall results with our method that are comparable to the results on pseudo-ambiguity data, and can improve the disambiguation in many cases.

3.4.5. Error analysis

We analyzed the samples that were classified incorrectly to see what went wrong. In this section, we address problems we found and how they can be partly fixed to improve the disambiguation accuracy.

Since we did not use gold labeled data for coreference but annotated our dataset automatically, there are some cases where mentions were incorrectly classified as coreferent and disreferent, respectively. This directly influences the quality of the r-contexts we extracted:

- **Incorrectly classified as coreferent:**
  Adds non-relevant context, which introduces noise.

- **Incorrectly classified as disreferent:**
  Misses relevant context, which might be necessary to make more reliable disambiguation decisions.

Coreference resolution is a task in natural language understanding which has yet to be solved (Ng, 2017). One of the hardest problems in automatic coreference resolution is resolving pronouns correctly, which has been addressed in previous work (e.g., Rahman and Ng (2012); Peng et al. (2015)). Proper names can mostly be identified with a string matching feature, e.g., *"Ehud Barak"*, *"Barak"*, and *"Mr. Barak"*. However, resolving pronouns is harder because there are often several possible candidate mentions the pronoun can be coreferent with due to them sharing same features (e.g., number and gender). Systems have included additional features to resolve pronouns, for example, a distance feature that assumes that the mention closest to the pronoun is the correct candidate (e.g., Soon et al. (2001)). While this solves some of the cases involving pronouns, there are still more complex cases which require world knowledge to solve them, as illustrated by Winograd (1972) in the following example:

(3.8) a. *"*$\boxed{\textit{The city councilmen}}_1$ *refused* $\boxed{\textit{the demonstrators}}_2$ *a permit because* $\boxed{\textit{they}}_1$ *feared violence."*

b. *"*$\boxed{\textit{The city councilmen}}_1$ *refused* $\boxed{\textit{the demonstrators}}_2$ *a permit because* $\boxed{\textit{they}}_2$ *advocated revolution."*

The two sentences in example 3.8 are very similar except for the verb and object phrase in the subordinate clause. Depending on the meaning of the subordinate clause, the pronoun *"they"* has a different antecedent in the main clause. In example 3.8a *"they"* refers to *"the city councilmen"* and in example 3.8b *"they"* refers to *"the demonstrators"*. A human reader can interpret the sentence and resolve the pronouns correctly because he or she has world knowledge of councilmen, demonstrators, and the semantic relations between them and the statement in the subordinate clause. A system cannot make informed decisions about these sentences without having that world knowledge.

Looking at our errors, examples 3.9 and 3.10 show sentences from two different contexts. It should be noted that they only show the sentences which are needed to understand the error, not the full context. In both cases, the possessive determiner *"his"* was incorrectly classified as coreferent with the wrong mention.

(3.9) a. *“Two weeks ago, police arrested a 26-year-old security officer assigned to guard* $\boxed{\textit{Ehud Barak}}_1$*, the leader of the opposition Labor Party, after a Shin Bet government security agent overheard the man saying that ‘someone should murder all the do-gooder Ashkenizi[sic] Jews, including* $\boxed{\textit{Ehud Barak}}_1$*.’ ”*

b. *“Confidential records of* $\boxed{\textit{Ehud Barak}}_1$*’s election strategies were stolen.”*

c. *“Since* $\boxed{\textit{his}}_1$ *election, for instance,* $\boxed{\textit{Prime Minister Benjamin Netanyahu}}_2$ *has repeatedly been called a ‘liar’ by* $\boxed{\textit{his}}_1$ *political foes.”*

(3.10) a. *“The United States scrambled on Monday to keep the October agreement between Israel and the Palestinians on track, with Secretary of State Madeleine Albright warning her Israeli counterpart,* $\boxed{\textit{Ariel Sharon}}_1$*, to adhere to the timetable set in the agreement and not to add new conditions.”*

b. *“Albright said at a news conference here with* $\boxed{\textit{Ariel Sharon}}_1$*, and she encouraged both sides to bring their differences, about the most recent agreement and everything else, to the negotiating table.”*

c. *“Albright dismissed questions about whether* $\boxed{\textit{linton}}_2$[sic] *might postpone* $\boxed{\textit{his}}_1$ *visit, saying that the visit itself is a part of the agreement.”*

The entity that needs to be identified is *“Ehud Barak”* as shown in sentence 3.9a. Sentence 3.9c is about another entity, *“Prime Minister Benjamin Netanyahu”*, and both possessive determiners *“his”* are coreferent with *“Netanyahu”*. However, the Standford coreference system incorrectly classified them as coreferent with *“Ehud Barak”*. This resulted in the sentence being added to the r-context, even though it has nothing to do with the entity we want to disambiguate. The same happened in example 3.10, where the entity to disambiguate is *“Ariel Sharon”* in 3.10a. Sentence 3.10c was extracted as relevant because the possessive determiner *“his”* was incorrectly classified as coreferent with *“Ariel Sharon”* instead of *“Clinton”*.

In sentence 3.9c the coreference system classified the possessive pronouns *“his”* as coreferent with the noun phrase *“Ehud Barak”* from he previous sentence (3.9b). Sentence 3.10c contains a typographical error (*“linton”* instead of *“Clinton”*). The Stanford tool still correctly classified *“linton”* as a named entity of the class `PERSON`, but did not classify it as coreferent with the possessive determiner. Instead, the possessive determiner was classified as coreferent with *“Ariel Sharon”* from the previous sentence.

The examples in 3.11 and 3.12 have the same problem as in the previous examples—a pronoun was incorrectly classified as coreferent with the entity that needs to be disambiguated (*“Ariel Sharon”* in sentence 3.11a and *“Joe Montana”* in sentence 3.12a).

(3.11) a. *"$\boxed{\textit{Ariel Sharon}}_1$, the Israeli foreign minister, has a unique view of the threat in the Balkans."*

b. *"$\boxed{\textit{It}}_1$ is not of a triumphant Serbia expelling Kosovars."*

(3.12) a. *"From, 'So when did you stop stalking $\boxed{\textit{Joe Montana}}_1$?' to 'Do you still wear that funny Mormon underwear?' "*

b. *"I don't think I've gotten more than a couple of questions about $\boxed{\textit{Montana}}_1$ since we played them in September."*

c. *"I'm sure $\boxed{\textit{it}}_1$'ll come up again next week."*

The difference to the examples before is that here the pronoun is the third-person, singular neuter pronoun *"it"*. The Stanford coreference system enforces agreement constraints for pronominal coreference resolution using several agreement features (number, gender, person, animacy, NER label) (Raghunathan et al., 2010). It is not clear why the system classified these pronouns as coreferent with our entities, considering the feature clash in gender (masculine vs. neuter). By doing this, sentences which are not about the entity were added to our context.

Another issue with the coreference system is shown in examples 3.13 and 3.14:

(3.13) a. *"[KEY ACQUISITIONS]. Bill Tobin (Chicago), vice president of football operations, linebacker Tony Bennett (Packers), running back $\boxed{\textit{Marshall Faulk}}_1$ (No. 1 draft pick), quarterback $\boxed{\textit{Jim Harbaugh}}_1$ (Chicago), $\boxed{\textit{Don Majkowski}}_1$ (Packers), $\boxed{\textit{Browning Nagle}}_1$ (Jets)."*

b. *"George and Trudeau are big losses whose talents weren't replaced by $\boxed{\textit{Harbaugh}}_1$, $\boxed{\textit{Majkowski}}_1$ and $\boxed{\textit{Nagle}}_1$."*

(3.14) a. *"Senator John F. Kerry Wednesday sharply criticized the conservative voting record of GOP vice presidential candidate $\boxed{\textit{Dick Cheney}}_1$, while acknowledging [...]"*

b. *"The criticism earned Kerry an instant rebuke from $\boxed{\textit{Senator John McCain}}_1$, the Arizona Republican who had joined Kerry at the press conference."*

c. *"$\boxed{\textit{McCain}}_1$ offered Kerry a near-endorsement and tribute as one of 'the best Americans in both parties,' but when Kerry criticized $\boxed{\textit{Cheney}}_1$, $\boxed{\textit{McCain}}_1$ complained: [...]"*

Sentence 3.13a is not a full sentence, but only consists of a list of people and some additional information about them, separated by commas. The coreference system could not process this sentence correctly due to the lack of a verb and possible objects. The result is that the system classified several mentions as coreferent with *“Marshall Faulk”*, thinking they were appositions for this entity. Therefore, our system extracted sentences which were about non-relevant entities as can be seen in sentence 3.13b.

This problem does not only occur in sentences which consist of lists only. In 3.14c the sequence *“Cheney, McCain”* was misinterpreted by the coreference system as an apposition when it is actually an object of the main clause (*“Cheney”*) and a subject of the subordinate clause (*“McCain”*). The syntactical parse tree and semantic dependencies were correctly analyzed by the Stanford tool, however, the coreference system still misclassified them. This resulted in the entity *“McCain”* being treated as coreferent with *“Dick Cheney”* throughout the entire document, adding all sentences which are about *“McCain”* (e.g., sentence 3.14b) to our context.

The above examples showed problems with sentences about other entities that were added to our context due to misclassifications of the coreference resolution system. These problems could probably be avoided by using a dataset with gold annotations because human annotators are less likely to make classification mistakes. However, in a real scenario, gold annotations are usually not available and thus, predicted labels by an automatic coreference resolution tool give more realistic results. Still, it is possible to improve the quality of the annotations to some degree:

- Going through the system annotations and manually correcting mistakes. This task is less expensive than manually annotating a dataset from scratch. However, it still takes a large amount of time to go through the data, identify the mistakes, and then correct them.

- Not relying on a single coreference system, but using a combination of coreference resolvers (“model stacking”) to obtain better results as was done in previous work (e.g., Björkelund and Farkas (2012); Clark and Manning (2015)).

The following examples illustrate that sentences which contain the personal pronoun *“I”* do not always contain helpful context:

(3.15) a. *“*$\boxed{\textit{Curtis Martin}}_1$ *doesn’t remember the exact number of times he watched the film of the Jets’ defense hold the Colts’ standout running back.”*

b. *“*$\boxed{\textit{I}}_1$*’m the type who doesn’t put any limits on myself.”*

c. *“*$\boxed{\textit{I}}_1$ *feel that there’s no limit.”*

d. *“*$\boxed{\textit{I}}_1$ *always want to improve.”*

(3.16) a. *“To* $\boxed{\textit{Peyton Manning}}_1$*’s surprise, he was greeted more civilly than that.”*

b. *"Asked to grade his own performance,* $\boxed{\textit{Manning}}_1$ *smiled and said: "*$\boxed{\textit{I}}_1$ *never do that."*

c. *"*$\boxed{\textit{I}}_1$ *'m sure you guys will do that."*

In the examples 3.15 and 3.16, the pronoun *"I"* was correctly classified as coreferent with the entities we wanted to disambiguate (*"Curtis Martin"* in 3.15a and *"Peyton Manning"* in 3.16a). The Stanford coreference system that we use includes a discourse processing sieve which matches speakers to pronouns (Lee et al., 2011). After identifying the speakers in a text, the system matches the pronouns *"I"* and *"you"* to the appropriate speakers by using sieve heuristics and constraints by subsequent sieves.

This discourse processing sieve seems to be working quite well in general. The pronouns in the examples 3.15 and 3.16 are also classified as coreferent with the correct mention. The basic intuition is that if an entity utters a sentence in which they are referring to themselves (or if another entity utters a sentence referring to the entity in question), then the sentence contains useful information because it is a closer description of the entity. However, in reality these sentences are often too general and could be said by several people as can be seen in examples 3.15b, 3.15c, 3.15d, and 3.16c, i.e., they do not provide distinct features that help disambiguate the entity. Adding such sentences to the context adds more noise.

This seems to be more of a problem with very short sentences that contain these pronouns. When investigating other sentences, it can be observed that longer ones tend to contain more useful information as seen in examples 3.17a–3.17f:

(3.17) a. *"I think the big reason we're the No. 1 defense is because of Casey."*

b. *"I don't think he's ready for the total package of the NBA."*

c. *"I think we are all working toward the goal of being professionals and I think this is my time to go after it with all my energies."*

d. *"Presumably I had not joined in the bestialization of the Serbs."*

e. *"Perhaps I had not supported the 'morality' of NATO's war."*

f. *"I wish he had been around to help protect my Albanian interpreter's family from Serb gunmen the night before."*

While longer sentences contain more information in general, there are some of these sentences where the information is more about other entities (e.g., in sentence 3.17b). These sentences can still add helpful information if the statements have a close relation to the entity which needs to be disambiguated. Possible solutions to improve the problem would be:

- Discard all coreferent pronouns *"I"* and *"you"* from the coreference chain so that the sentences they occur in are not added to the r-context. This is the most restrictive solution which might miss many sentences that are actually providing helpful information.
- Include the coreferent pronouns *"I"* and *"you"* but impose a restriction on the sentence length and only add sentences which are longer than what the restriction requires.

Another problem we observed was that in some cases both entities of an entity pair occurred in the same document. In some of these documents, both entities occurred together in at least one sentence. In general, the context we create for both entities will still be different because we have different coreference chains for each entity. However, the more sentences both entities share, the more information about them is mixed together, which is confusing for a classifier.

Examples 3.18 and 3.19 show sentences from two documents mentioning both *"Ehud Barak"* and *"Ariel Sharon"*:

(3.18) a. *"Barak has been holding negotiations to form a 'national emergency government' with Ariel Sharon, leader of the right-wing opposition Likud Party."*

b. *"An alliance with Sharon would insulate Barak, at least temporarily, from a possible opposition attempt to bring down his government when the Israeli Parliament, called the Knesset, reconvenes on Monday."*

c. *"Barak and Sharon continued haggling Friday over how they might share power in a joint government."*

(3.19) a. *"He also said the Palestinians hoped to resume the talks after the Israeli elections, even if it is with Barak's opponent, Ariel Sharon of the right-wing Likud Party."*

b. *"As such, the Palestinians, who are alarmed at the prospect of Sharon becoming prime minister, gave Barak a last-minute hand."*

c. *"End-of-the-week polls, in which Sharon maintained his commanding lead over Barak, showed that if the Taba talks resulted in an agreement, it would cost the prime minister support."*

Sentence 3.18a consists of information about *"Barak"* in the first half and information about *"Sharon"* in the second half. Sentence 3.18b is mostly about *"Barak"*. In sentence 3.18c all the information is about both entities at the same time. A mix of information can also be seen in sentences 3.19a–3.19c.

The bag-of-words model we use for classification performs well in general. However, these examples show the limitations of the BOW model. A BOW model does not capture word order in a sentence or relations between words (e.g., *"X of the right-wing Likud Party"*). Due to these restrictions, the sentences in 3.18 and 3.19 look the same for both entities. To resolve this problem, a more sophisticated model is necessary which can capture more than the word frequency of the BOW model, for example:

- Relation between a verb and its arguments (e.g., *"Barak has been holding negotiations with Ariel Sharon"*)
- Relation between a noun and its arguments (e.g., *"Ariel Sharon of the right-wing Likud Party"*)

3.5. Related work

Related work on named entity disambiguation and closely related tasks has made use of both real data (e.g., Cucerzan (2007); Milne and Witten (2008); Ratinov et al. (2011); Hoffart et al. (2012); Röder et al. (2014)) and pseudo-ambiguity data (e.g., Pedersen et al. (2005, 2006); Kulkarni and Pedersen (2005)). For a more detailed overview of the datasets used in other work we refer to our discussion in Section 3.3. We give a general overview of NED and related tasks in Chapter 2, Section 2.2. In the remainder of this section we discuss work that is related to the approach we used in this chapter in two ways: (i) approaches using pseudo-ambiguity data and (ii) approaches that use specific context sizes around the entity.

Mann and Yarowsky (2003) create syntactic and lexical patterns to extract biographic facts (e.g., birth date, birth place, and occupation) from English and Spanish texts. They use these biographic facts as well as the most relevant words in the document collection to disambiguate proper names with an unsupervised clustering technique. For their experiments they create a pseudo-ambiguity dataset. They collect up to 1000 websites for each of their eight target proper names and take a maximum of 100 pages for each person. They then randomly choose pairs consisting of these eight target persons and obtain a total of 28 pseudo names. They do not provide detailed results of their experiments on this pseudo-ambiguity data but only an overall disambiguation accuracy over all 28 pseudo names. They also perform experiments on a hand-labeled dataset of real ambiguous persons; however, they only evaluate the two major sense clusters.

Pedersen et al. (2005) use a method of clustering similar contexts to identify proper names. The contexts are created by taking about 25 words on each side of an ambiguious proper name. From these contexts they identify significant bigrams as features, where a bigram can have one intermediate word inbetween, and use these bigrams to create a matrix, which is further reduced by singular value decomposition to create second order context vectors. They use a repeated bisections approach for clustering and experiment with smaller and larger training and test scopes around the ambiguous name. They conduct their experiments on six pairs of pseudo names (persons, organizations, nations, countries), but they do not apply their approach to real ambiguous data. Pedersen et al. (2006) take the same methodology from their previous paper and apply it to languages other than English—namely, Bulgarian, Romanian, and Spanish—and show supporting fact that their method is language independent.

Kulkarni (2005)'s work is very similar to (Pedersen et al., 2005) but they do not only consider ambiguity between two entities but introduce a cluster labeling technique for disambiguating multiway distinctions. Kulkarni and Pedersen (2005) extend on their previous work and use their approach for disambiguating names and email clustering, where they treat email messages as contexts.

Cross-document coreference determines whether mentions of named entities in different documents refer to the same real world entity or not. Since different entities with the same name can appear in different documents, CDCR needs to be able to disambiguate such entities, to determine whether they are coreferent or not. Similar to our work is the approach by Bagga and Baldwin (1998b). They use within-document coreference to first identify all noun phrases that are coreferent with a given entity. They then create small document summaries with all sentences that contain these noun phrases and cluster the documents based on these summaries. They do not experiment with different context sizes to find out whether using all sentences for the summary improves the disambiguation or introduces more noise. They evaluate their approach on a small hand-labeled corpus consisting of 173 articles about 35 different entities with the name *"John Smith"*, but they do not perform evaluation on other names.

Gooi and Allan (2004) base a large part of their work on (Bagga and Baldwin, 1998b) and perform experiments on the *John Smith corpus* created by Bagga and Baldwin (1998b) as well as a pseudo-ambiguity dataset they create. They obtain context to disambiguate entities in two steps. First, they take a fixed window of 55 words around each mention, with the mention being in the middle of the context. Second, for each occurrence of the entity in the document, they merge these small contexts into one large context. They create the *Person X corpus*, a pseudo-ambiguity dataset where they replace occurrences of proper names of the form *"firstname lastname"* with *"person-x"*. However, they filter out all cases that consist of only one word (e.g., *"John"*). The persons they use for their pseudo-ambiguity dataset are not specifically chosen to have similar backgrounds but include many different persons with a large variety of backgrounds.

Different context sizes have been used and investigated in related work. Some approaches use the entire document to obtain the maximum of information. Other approaches assume that the most important information is close to the entity and use a fixed window of words around the entity.

Bunescu and Pasça (2006) exploit information from Wikipedia, such as redirect pages, disambiguation pages, categories, and hyperlinks, to train a support vector machine for disambiguating named entities. As context they use a fixed window size of 55 words around the entity, which they take from the work done by Gooi and Allan (2004). Likewise, based on this window size is the context used by Nguyen and Cao (2008), who identify named entities in text and link them to Wikipedia. They use an incremental approach that identifies the best candidates in several rounds while using information gained in previous rounds for improved identification in further rounds.

Bollegala et al. (2006) perform experiments on pseudo-ambiguity data and real data. For the pseudo-ambiguity data they take three persons, collect 50 documents per person from the web, and then replace every name mention with *"person-x"*. For the real data they choose four ambiguous names and collect 50 documents per name. Their approach is based on the assumption that each of these documents they collected is representative for the corresponding name. They use the entire document as the context and cluster the documents using a group-average agglomerative clustering algorithm. The output of their algorithm are phrases extracted from the documents that describe the entity.

Ikeda et al. (2009) present an algorithm for disambiguating proper names in web search results. They implement a two-stage clustering algorithm that uses several features (named entities, compound keywords, and URLs) and apply it on the WePS-1 and WePS-2 datasets. They experiment with different window sizes for the feature extraction: 50, 100, and 200 words, as well as taking the whole document. They found that increasing the window size gives better results for both of their evaluation measures (B^3 F_1 score and Purity/Inverse Purity F_1 score[38]). However, taking the entire document only increases the results for the B^3 F_1 score while it worsens results for the Purity/Inverse Purity F_1 score.

Hoffart et al. (2011) combine several approaches and measures that have been suggested in earlier work (prior probability of an entity, context similarity, and candidate coherence). For the context creation they take the entire document but discard stopwords and the mention itself. They perform experiments that give tokens which are closer to the mention more weight than tokens which are farther away. However, they found that this did not help improve the results. They do not explain in detail how they determine the weights and how it changes the results.

[38]Details on these scores can be found in (Artiles et al., 2007).

Han et al. (2011) develop a graph-based method to link named entities to their corresponding entities in a knowledge base. Nodes represent both mentions of named entities in a text as well as the entities in the knowledge base. Edges between mention nodes and entity nodes denote a compatible relation between them and are weighted based in the probability of their relation. They use a fixed window size of 50 around the mention as their context, basing it on work done by Pedersen et al. (2005).

Li et al. (2013) also use knowledge bases to disambiguate entities and to link them to their corresponding KB entity. They note that knowledge bases will probably never be complete, which is a problem for identification. They propose an algorithm that does not only rely on the information in the knowledge base but additionally extracts information across documents. As context size they choose a window of 60 words around the entity, however, they do not explain why they chose this specific context size.

In recent years, approaches have shifted more and more from local models which mostly use contextual features (i.e., using the context around the entity, e.g., as bag-of-word or as concept vectors; e.g., Bunescu and Paşca (2006); Mihalcea and Csomai (2007); Ratinov et al. (2011); Lazic et al. (2015); Eshel et al. (2017)) to global models which add more sophisticated features such as entity-relatedness features (e.g., Cucerzan (2007); Milne and Witten (2008); Han and Zhao (2009); Cheng and Roth (2013); Guo and Barbosa (2014b); Pershina et al. (2015); Globerson et al. (2016)). For a short overview of local vs. global approaches we refer to Chapter 2, Section 2.2.2. However, contextual features are still important for disambiguating entities and local models can be preferred depending on the use case. Eshel et al. (2017) point out that for text documents that are short and noisy (e.g., social media content or questions and answers) it is hard to apply global models and local models can perform better on this type of text. They use a context window of at most 20 words to each side of the entity which they feed into an Attention-RNN model that performs better than various baseline models.

Another change that has happened in recent years is the increase of models that use neural networks to disambiguate named entities. Example work includes stacked autoencoders (e.g., He et al. (2013); Lubani and Mohd Noah (2018)), convolutional neural networks to capture the relation between context, mentions, and candidates (e.g., Sun et al. (2015); Francis-Landau et al. (2016)), or recurrent neural networks that exploit the sequence of text in a document (e.g., Eshel et al. (2017); Mueller and Durrett (2018)). Neural networks can exploit a wide range of features, and contextual features as well as the context size are still important for neural networks.

Neural network approaches show good performance in general for disambiguating named entities, but the models still fail to disambiguate entities in some situations. For example, Mueller and Durrett (2018) point out that Eshel et al. (2017)'s model has trouble identifying examples that are easy to disambiguate for a human reader. They claim that for noisy texts with limited context windows, feeding the network with standard representations of context is insufficient and that more features are necessary.

Many neural network approaches use the entire document as the context and let the weights in the network decide which parts are important for disambiguating the entity (e.g., He et al. (2013); Francis-Landau et al. (2016)). Sil et al. (2017) argue that choosing the entire document is not useful based on their experiments, and propose to use a sentence-based context related to the entity. They use an approach that is similar to our approach in this chapter. First, they determine the coreference chain of an entity and use this information to concatenate all sentences that contain the entity. Then, they feed these contexts into a convolutional neural network that produces fixed-size vector representations from the input. To evaluate the model, they create context representations from the first paragraphs of Wikipedia articles as possible target entities. For a fine-grained context modeling, they choose a context window of four words to each side of the entity, and use long short-term memory (LSTM) networks on them. Their model outperforms previous models that use the entire document as context.

3.6. Summary

In this chapter we explored the usefulness of coreference chains for improved identification of proper names. We introduced a type of context (*r-context*) which differs from standard approaches that take the entire document or a fixed window around an entity as the context. To obtain r-context, we identified all coreferent expressions of an entity in the document, and combined the sentences in which these expressions occur to one context. We also applied different filters based on POS tags to obtain different types of coreference-based context.

After preliminary experiments we found that more data is needed to evaluate our approach. Since creating a large dataset manually is an expensive task, we decided to create a pseudo-ambiguity dataset. We conducted several experiments with different parameters to study how they influence the results. To demonstrate that pseudo-ambiguity data can be used for our task we conducted similar experiments on a small dataset using real data (manually annotated for named entities) and compared the results to the results obtained with pseudo-ambiguity data. Our results demonstrate that pseudo-ambiguity data is a useful resource to test our proposed method.

We showed that using r-context improves disambiguation results compared to using a fixed context around the entity or the entire document. Finally, we discussed some errors and limitations and possible ways to improve them.

4. Improved disambiguation of little-known named entities

4.1. Motivation

In the previous chapter (Chapter 3) we started our exploration of improving entity retrieval by developing an approach that improves disambiguation of proper names in general. With our approach we followed the standard way of named entity disambiguation, viewing the task as a supervised classification problem (e.g., Han et al. (2004); Dredze et al. (2010); Yamada et al. (2016)). These approaches have a small drawback: they need training material about the persons to be able to disambiguate them in unseen text. This is generally not a problem for entities which are well-known and for which much training material can be collected. However, for little-known entities, on which we focus in this chapter[39], there might not be enough data.

As we have shown in the introduction chapter (Chapter 1, Section 1.2), most search requests in a search engine like Google are made for the most well-known persons. Such statistics might explain in part why previous approaches usually use medium to high frequency name/referent combinations in their training and testing scenario. Even when neglecting little-known referents, a system can still achieve a high accuracy for the returned results because the non-prominent mentions are orders of magnitude less frequent and many existing datasets are biased towards well-known entities (Ilievski et al., 2018). Furthermore, it is much easier to obtain suitable training material for medium to high frequency referents, for example, by scraping the web for documents about them or using a knowledge base such as Wikipedia as a source. To be included as an article in Wikipedia, a person needs to be "worthy of notice", which means the person is quite known and it is likely to find more information about them on the web.

In practice, however, any system that is systematically overlooking theoretically identifiable people is problematic. For example, search engines and question answering systems will return more unwanted or incorrect results, or not find results about the

[39]The approach presented in this chapter was published in (Glaser and Kuhn, 2016).

person at all because of an incorrect mapping to a similar, more well-known person.

Besides these medium to high frequency named entities, i.e., the “prominent” bearers of a name, there are usually many non-prominent bearers of that name. Sarmento et al. (2009) note the high skewness of distributions of mentions on the web, i.e., that there are usually one or two very prominent bearers of the name and most of the mentions found in documents refer to those entities. For the non-prominent entities it is hard to find enough suitable training material by harvesting the web and they often do not appear in the major reference knowledge bases. For these reasons, these cases about non-prominent name bearers have received little attention in previous research work.

$$\begin{tikzcd} & \boxed{\textit{Michael Jackson}} \arrow[dl] \arrow[d] \arrow[dr] & \\ \text{singer} & \text{writer} & \dots \end{tikzcd}$$

Figure 4.1.: Possible real world referents for the proper name *“Michael Jackson”*.

Figure 4.1 shows possible real world referents for the proper name *“Michael Jackson”*. The most prominent person with the name *“Michael Jackson”* is the American singer. There are medium prominent referents such as the Canadian actor or the English writer and beer expert. Some of the the referents that appear in Wikipedia are less prominent, for example, the Irish bishop.

Besides these referents that appear in Wikipedia, there are many other people with the name *“Michael Jackson”*. For example, there is a *“Michael Jackson”* who is a Canadian law professor, but is not among the 35 or so Michael Jacksons with their own Wikipedia articles. However, ways to establish a unique reference to the correct entity exist beyond major knowledge bases, for example, using name authority files curated by national libraries. Examples for providers of such name authority files are OCLC’s WorldCat Identities[40], the Virtual International Authority File[41] (VIAF), or the Integrated Authority File[42] (GND; “Gemeinsame Normdatei”). It is also possible to use other lists that provide unique identification from a specific application perspective, such as staff lists on institutional websites or listings of sportspeople.

Table 4.1 lists a small selection of different real world entities with the proper name *“Michael Jackson”* and whether or not they have an article or entry in Wikipedia, DBpedia, WorldCat, and VIAF. The prominent name bearers, such as the American singer, the British beer expert, and the Canadian actor are represented in the big knowledge-

[40]`https://www.worldcat.org/identities/`
[41]`https://viaf.org/`
[42]`https://www.dnb.de/gnd`

Real world entity	Wikipedia	DBpedia	WorldCat	VIAF
American singer	✓	✓	✓	✓
Canadian actor	✓	✓	✓	✓
British beer expert	✓	✓	✓	✓
⋮				
Canadian professor	✗	✗	✓	✓

Table 4.1.: Examples of different entities with the name *"Michael Jackson"* and their occurrence in knowledge bases and authority files.

bases like Wikipedia and DBpedia. The Canadian law professor does not have an article in Wikipedia. Since DBpedia uses data from Wikipedia, the entity is not in this knowledgebase either. However, the entity can be found in WorldCat with the unique identifier `lccn-n84079060`[43] and in VIAF with the unique identifier `43238033`[44]. WorldCat provides additional information about each entity, for example, alternative names, profession (role), works created by the person, genres, languages, related entities, and associated subjects weighted by how important the relevance is to the given entity. VIAF provides similar information about entities.

In this chapter, we focus on these non-prominent name bearers and propose a system that can identify them. The approach for the system rests on the idea that if people share the same properties (such as profession and nationality), they appear in contexts which are quite similar topic-wise. For example, the topics of a text about a person who is a singer will be similar to the topics of texts about other singers, but less similar to topics of texts about politicians or actors. Furthermore, the idea is that even if there is no or only little training material available for a non-prominent name bearer, it is still possible to obtain some properties from name authority files or other listings, such as profession, nationality, and institutional affiliation.

Knowing properties that people can have, we can collect textual material about different people with the same properties. This material can then be used as a proxy to identify an unknown person with the same properties, without the need to have specific training material for this person. For example, to disambiguate *"Michael Jackson, the Canadian law professor"*, we can use textual material about people with the same properties, i.e., mentions of all Canadians, of all law scholars, professors etc.

Example 4.1 shows a small text snippet about *"Michael Jackson, the Canadian law professor"*. While the profession is not mentioned explicitly in the text, there are several words which have a high probability to appear in topics related to the profession,

[43]`https://www.worldcat.org/identities/lccn-n84079060/`
[44]`https://viaf.org/viaf/43238033/`

such as *"teaching"*, *"courses"*, and *"law school"*. While texts about other professors etc. might have different words, many of them will be associated with topics related to the profession. This topic information can be used to identify the unknown entity.

(4.1) **Michael Jackson** has been involved in teaching human rights for over thirty years. His courses on these subjects were the first to be introduced in a Canadian law school.

In summary, we make the following contributions in this chapter:

- We demonstrate that it is possible to identify little-known entities (for which however certain properties are known) without having explicit training material for them, by using some aggregate of textual data that shares the same properties.
- We show how to create this aggregate of text data for several properties, as well as how to create a silver standard corpus that can be used for evaluation.
- We perform a systematic study with different parameters (corpus type, context size, number of topics used for the topic model) and present detailed results which show the effects of these parameters.
- We show that our approach is indeed very helpful for disambiguating (i) entities for which not much training material is available, and (ii) also for entities with little surrounding context, both of which are useful for many applications.

The rest of the chapter is structured as follows. In Section 4.2 we introduce our system that learns properties of persons and uses these properties to disambiguate unseen persons for which no specific training material is available. In Section 4.3 we first describe the different datasets we create for our experiments in detail. We then discuss the results that we obtained in the experiments, including an investigation of the parameters that we used. Section 4.3.4 presents an error analysis. We describe related work in Section 4.4. At the end of the chapter, we give a summary in Section 4.5.

4.2. Methods

In this section, we describe how our system works. The underlying idea of the system is that it does not have to collect textual training data for a specific person (e.g., the Canadian law professor *"Michael Jackson"*) by using corpora or scraping websites and mining information about the person. Instead, the system makes use of the fact that people share some properties with other people, such as nationalities and professions.

This means, we can use some aggregate of the textual material about different people that have the same properties. In the case of *"Michael Jackson, the Canadian law professor"* this textual material would consist, for example, of mentions of Canadians, law scholars, and professors etc. Thus, our system is independent from existing training data or obtaining training data through other means (e.g., web scraping), and can be applied to any text without the preliminary step of extracting specific information about the entities in the text.

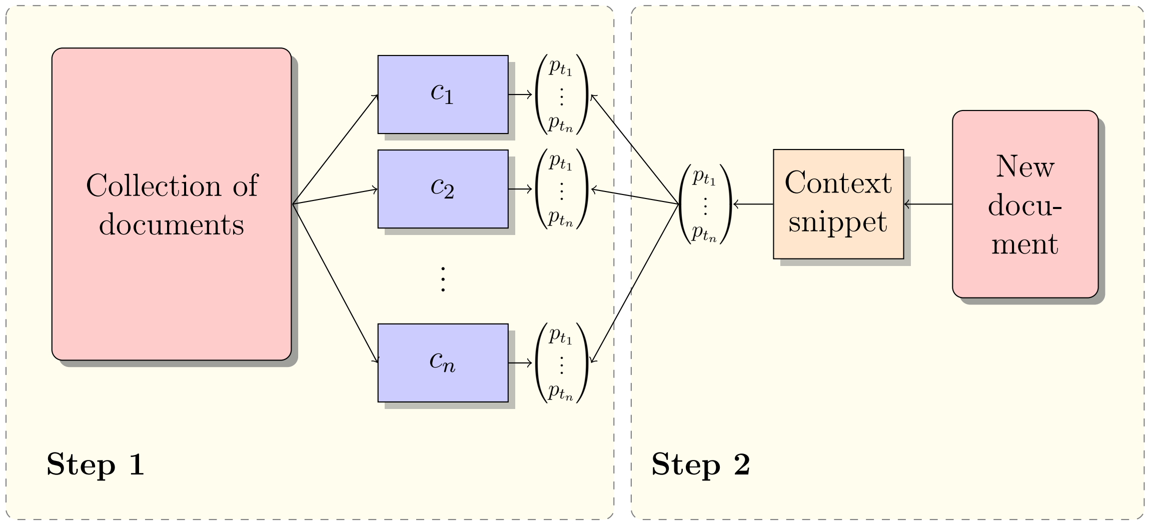


Figure 4.2.: System that learns characteristics of people and uses them to disambiguate unknown people.

Figure 4.2 shows our system, with its process divided into two steps. In the first step, we take a collection of documents and extract documents with specific properties, e.g., documents about singers, authors, and other professions, and documents about Americans, Canadians, and other nationalities. We then concatenate these extracted documents to individual corpora $c_1, ..., c_n$ which we call *properties corpora*. After this, the topics for these properties corpora are determined by using a topic model. In the second step, our system is applied to new unknown proper names. This is done by determining the topics for the proper name based on the chosen context and then comparing these topics with the topics of our properties corpora to find the ones with the greatest similarity. We describe both steps in more detail in the next two subsections.

4.2.1. Learning properties of persons

In the first step (left side of Figure 4.2), our system takes a collection of documents as the input and learns characteristic properties that people can have. Examples of such properties are:

- Nationalities (American, Canadian, German, Irish, Japanese, ...)
- Professions (singer, author, president, tennis player, ...)
- Affiliations (university, company, ...)
- Engagements (organization, charity, ...)

These properties are helpful to varying degrees. For example, every person has a nationality, i.e., this property is very helpful to identify a person. Most people also have some kind of profession (if not, they can be treated as "unemployed" or "no profession"). Affiliations are useful to disambiguate people with the same profession and nationality (e.g., two American linguistics professors from two different universities). Engagement is not a general property because not every person is engaged in something.

In our work we focus on two properties: nationalities and professions. We chose these properties because they are the most prominent properties people have, i.e., they are usually known. Further properties, such as affiliations, engagements or even more specific properties, can be easily integrated into the system at a later point of time.

Learning properties works as follows. The system takes a document collection (e.g., Wikipedia) as the input. From this document collection it extracts documents with the specific properties. For example, the system extracts documents about people who are singers, actors, and other professions, as well as documents about people whose nationality is American, Canadian, etc. In our pilot study we worked with a fixed set of entities that were all associated to one nationality and one profession. For each entity, we created one corpus that consists of extracted documents that belong to the respective properties. For example, if we know that an entity is American and a singer, we randomly choose (i) n documents about people who are American, and (ii) n documents about people who are singers, and concatenate them all into one "American singer corpus". We call these corpora we create *properties corpora* as each of them consists of documents with specific properties. These properties corpora are the basis for disambiguating unknown entities. The extraction process for the properties corpora is described in more details in Section 4.3.1.

These properties corpora are not very helpful in this form yet. Besides the useful properties, the corpora also contain much information that has nothing to do with these properties. Noisy data can interfer with the disambiguation process and result in worse results. To fully exploit the usefulness of these properties corpora, the actual information about the properties needs to be filtered out. To obtain this relevant information from the corpora we use topic models.

Topic models need to be trained before they can be applied on some data. We train topic models on different training collections to study whether the training data influences the quality of the model. A detailed description of the different training collections is presented in Section 4.3.1. After the topic models are trained, they can be applied to the properties corpora to obtain the relevant topic information. For example, the American singer corpus contains many words related to a singer, for example, album, music, and concert. The topic model determines a topic related to these words and assigns a high probability to the topic, given the corpus. On the other hand, topics about other words that are not related to the property, have a lower probability.

4.2.2. Using properties to disambiguate persons

In the second step (right side of Figure 4.2), our system is applied to new unknown proper names. To disambiguate a new person, the system needs some context around the person (e.g., a sentence or a paragraph). We call this extracted context a *context snippet*. The idea is to extract properties, i.e., topic information, from these context snippets and compare this topic information with the topic information from the properties corpora to find the most similar corpus.

We use the same trained topic models as in the first step (Section 4.2.1) and apply them to the context snippets to obtain topic information from them. By using the same topic models, the extracted topic information will be similar for both the properties corpora and the context snippets.

The topic information we obtain from a document (e.g., a properties corpus or a context snippet) can be represented as a vector of length n, where n is the number of topics. Each entry p_{t_i} in the vector corresponds to the probability of the topic t_i given the document.

We then compare the topic vector of the new person with the topic vectors of each properties corpora to find the corpus that is most similar to the context the person occurs in. For this comparison we use cosine similarity. Let $\mathbf{x}$ be the vector of the new person and $\mathbf{y}_c$ be the vector of the properties corpus c_j. The cosine similarity $\cos(\theta)$ between the two vectors $\mathbf{x}$ and $\mathbf{y}_c$ is defined as:

$$\cos(\theta) = \frac{\mathbf{x} \cdot \mathbf{y}_c}{||\mathbf{x}|| \cdot ||\mathbf{y}_c||} = \frac{\sum\limits_{i=1}^{n} x_i y_i}{\sqrt{\sum\limits_{i=1}^{n} x_i^2}\sqrt{\sum\limits_{i=1}^{n} y_i^2}} \tag{4.1}$$

$$\text{with } \mathbf{x} = \begin{pmatrix} p_{t_1} \\ \vdots \\ p_{t_n} \end{pmatrix} \text{ and } \mathbf{y}_c = \begin{pmatrix} p_{t_1} \\ \vdots \\ p_{t_n} \end{pmatrix}$$

where p_{t_i} is the probability of topic t_i given the document.

4.3. Experiments

In this section we describe the experiments we conducted. In Section 4.3.1 we first describe how we created different datasets for training our topic models, as well as training and test sets for the experiments. We then present the experimental setup in Section 4.3.2 with the tools and parameters we used, and describe the baselines. In Section 4.3.3 we first show and discuss the results from our experiments in general (precision, recall, F_1 score). Then, we investigate the parameters we used in more detail. We conclude with an error analysis in Section 4.3.4.

4.3.1. Data and preprocessing

Since our task uses a novel approach to disambiguate referring expressions in text, existing datasets for named entity disambiguation are not suited for the task. Existing datasets consist of data about specific people rather than data about general properties. In addition, they lack the annotations that are necessary to evaluate our approach. For these reasons we created our own data sets. For each part of our system we need different types of data:

- Data for training our topic models
- Data for creating properties corpora (training data)
- Data for evaluating our system (test data)

For training our topic models we use general data so that the topic models can learn a great variety of topics. Thus, we use Wikipedia as the main corpus for training our topic models. To compare it with a different corpus we also train our topic models on a corpus which mostly consists of newswire text.

The properties corpora consist of documents about people that share the same properties. We use Wikipedia as a source for the properties corpora because it provides metadata that can be used to extract documents with specific properties. These properties corpora are used as our training data.

Creating and manually annotating a test set for evaluation is expensive. The first step is to find sufficient data (e.g., using existing corpora or scraping the web for data). This step takes longer if the data needs to be more tailored to a specific task. Depending on the source, the data needs to be cleaned (e.g., removing HTML tags) and preprocessed (e.g., tokenization, POS tagging, parsing). Then, annotation guidelines need to be set up and annotators need to be trained. For a dataset of a big enough size this is very time consuming and expensive. It also requires some knowledge about the persons in the test set to link them to their correct real world entity correctly. This is especially hard when the test set should contain less-known name bearers as in our case; annotators are likely to make more mistakes or disagree with other annotators if they do not know these people.

To circumvent these problems we decided to create a test set automatically by exploiting the link structure in Wikipedia. We call this test set *silver standard test corpus.* This way, we can obtain a reasonable amount of data for evaluating our system without having to go through all the steps of annotating it. Entities need to be well-known to a certain degree to have their own Wikipedia article. Therefore, our test set consists of well-known persons. However, we can simulate the situation of non-prominent name bearers by leaving documents about them out of the actual training data, which turns the entities in our test set into non-prominent name bearers.

We use two data collections in this work:

- English Wikipedia, February 3, 2014 version[45]
- English Gigaword corpus (Graff and Cieri, 2003)[46]

In the following sections we describe the data used in our system in more detail. We also describe how we create the data.

Corpora for training topic models

We trained different topic models on different collections and parts of collections to study the effects of different sources.

- **Wikipedia - all articles (W$_{all}$)**
  As a general internet encyclopedia which can be edited by everyone, Wikipedia provides a great variety of articles. For this model we used all articles that are available in this Wikipedia version without any restrictions.

[45]Wiki dump enwiki-20140203

[46]We describe the English Gigaword corpus in more detail in Chapter 3, Section 3.4.1.

- **Wikipedia - articles of the category *living people* ($\mathbf{W}_{lp}$)**
  Wikipedia articles are classified into different categories. We built a model that only uses articles from the category *living people* to see if the properties we can learn from articles about people are more helpful for identifying new people than the properties we can learn from the entire Wikipedia.

- **Wikipedia - individual sections of articles of *living people* ($\mathbf{W}_{lps}$)**
  Many Wikipedia articles are very long and separated into several sections that are often very different with respect to their topics (e.g., *"early life", "career"*). We want to investigate if we can obtain more helpful topics by using individual sections that are about specific topics as opposed to taking entire articles that consist of many different topics.

- **English Gigaword - section *nyt* ($\mathbf{G}_{nyt}$)**
  The English Gigaword corpus consists of newswire text data in English. With this model we want to analyze if a newswire corpus provides different topics than an encyclopedia. We only use one part of the English Gigaword corpus for this model, newswire data from the source *The New York Times Newswire Service* (*nyt*).

Collection	# Documents
W_{all}	4,370,653
W_{lp}	647,659
W_{lps}	1,699,230
G_{nyt}	1,298,498

Table 4.2.: Number of documents for each model.

Table 4.2 shows the number of documents we used for each model. The entire Wikipedia consists of 4,370,653 documents, which makes the largest collection for our models. Filtering articles by category and only taking articles of the category *living people* gives us 647,659 documents, which is 14.82% of all articles in Wikipedia.

Dividing these articles into individual sections results in 1,699,230 documents. Famous people often have many sections, for example, ten sections for *"Michael Jackson, the American singer"* and nine sections for *"John Williams, the composer"* (including the introduction section; excluding sections for references, notes, bibliography, and external links). The structure of articles about people varies in general. Some people have fewer sections but more subsections, or have merged sections which are separated in other articles. For example, *"Angela Merkel"* has ten sections with early life and early political career in separate sections. *"Barak Obama"* has only five sections in total, early life and early career being merged in one section and with four subsections. We do not divide the articles into subsets as the subsets usually belong to one topic and are often rather small. Less known people generally have fewer sections because not much is known about their life or the article is less detailed. For example, articles about actors and singers often only list a filmography or discography but no other information. Many articles about less known people only consist of the introduction section with no additional sections. The average number of sections in articles of the category *living people* is 2.62.

The subset we use from the English Gigaword corpus consists of 1,298,498 documents. This is 29.71% of the size of the entire Wikipedia corpus. Each document corresponds to one news article and we do not divide them into individual parts.

Collection	# Topics					
	100	1,000	2,500	5,000	7,500	10,000
W_{all}	✓	✓	✗	✗	✗	✗
W_{lp}	✓	✓	✓	✓	✓	✓
W_{lps}	✓	✓	✗	✗	✗	✗
G_{nyt}	✓	✓	✓	✗	✗	✗

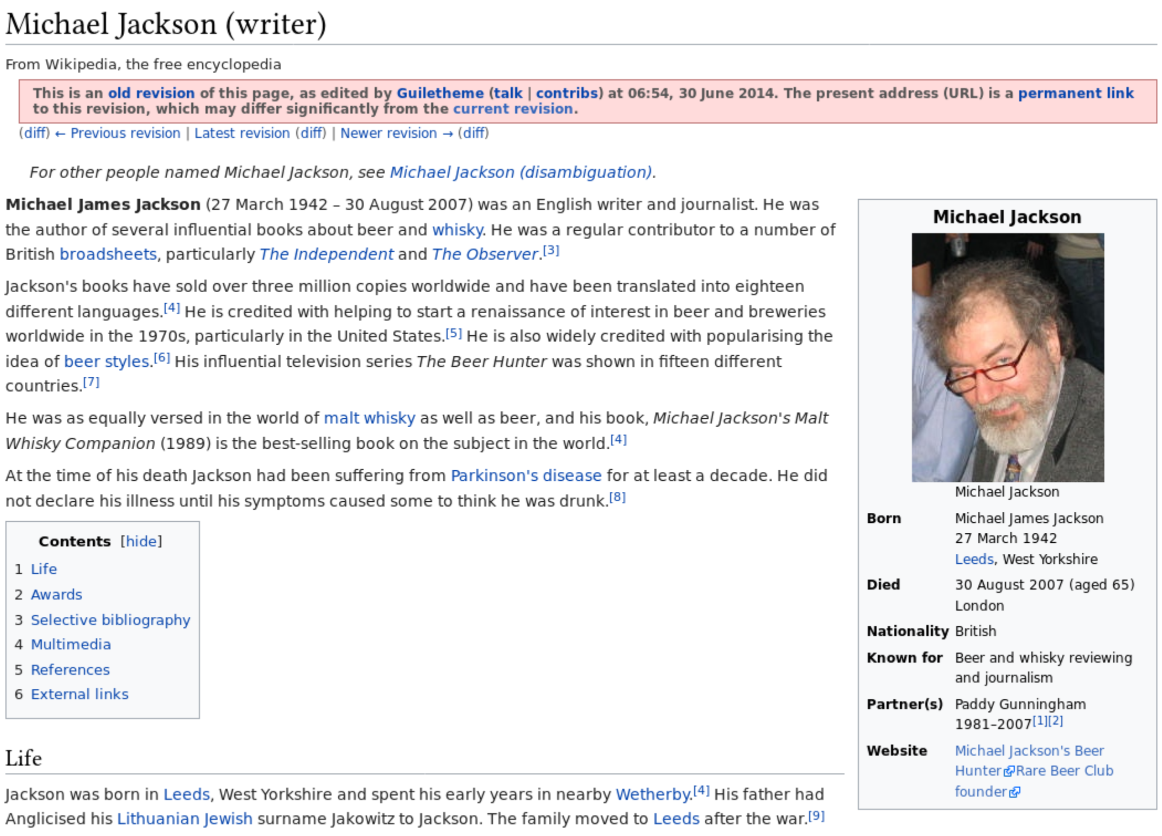
Table 4.3.: Models used for experiments.

We experiment with the number of topics for the topic model and create different models for each of our collections. The number of topics ranges from 100 (more coarse-grained topics) to 10,000 (more fine-grained topics). Table 4.3 shows the numbers of topics we used for the different collections. Due to the large number of documents in most collections, we only experiment with the full range of number of topics for the smallest collection (W_{lp}). We restrict the larger collections to a maximum of 1000 topics (Wikipedia) and 2500 topics (English Gigaword) for efficiency reasons.

By experimenting with different numbers of topics we want to investigate if more fine-grained topics will give us better results when disambiguating new people, or if a smaller number of topics is enough for the task.

Properties corpora

In this work we focus on the most prominent properties people have, nationalities and professions. To create properties corpora we extract Wikipedia articles about people that share these properties. Wikipedia provides information about nationalities and professions for most people listed in their database. If the nationality or profession is not known, the information is not available in the article.



Michael Jackson (writer)

From Wikipedia, the free encyclopedia

This is an old revision of this page, as edited by Guiletheme (talk | contribs) at 06:54, 30 June 2014. The present address (URL) is a permanent link to this revision, which may differ significantly from the current revision.

(diff) ← Previous revision | Latest revision (diff) | Newer revision → (diff)

For other people named Michael Jackson, see Michael Jackson (disambiguation).

Michael James Jackson (27 March 1942 – 30 August 2007) was an English writer and journalist. He was the author of several influential books about beer and whisky. He was a regular contributor to a number of British broadsheets, particularly *The Independent* and *The Observer*.[3]

Jackson's books have sold over three million copies worldwide and have been translated into eighteen different languages.[4] He is credited with helping to start a renaissance of interest in beer and breweries worldwide in the 1970s, particularly in the United States.[5] He is also widely credited with popularising the idea of beer styles.[6] His influential television series *The Beer Hunter* was shown in fifteen different countries.[7]

He was as equally versed in the world of malt whisky as well as beer, and his book, *Michael Jackson's Malt Whisky Companion* (1989) is the best-selling book on the subject in the world.[4]

At the time of his death Jackson had been suffering from Parkinson's disease for at least a decade. He did not declare his illness until his symptoms caused some to think he was drunk.[8]

Contents [hide]

Life

Jackson was born in Leeds, West Yorkshire and spent his early years in nearby Wetherby.[4] His father had Anglicised his Lithuanian Jewish surname Jakowitz to Jackson. The family moved to Leeds after the war.[9]

Michael Jackson

Michael Jackson

Born	Michael James Jackson 27 March 1942 Leeds, West Yorkshire
Died	30 August 2007 (aged 65) London
Nationality	British
Known for	Beer and whisky reviewing and journalism
Partner(s)	Paddy Gunningham 1981–2007[1][2]
Website	Michael Jackson's Beer Hunter Rare Beer Club founder

Figure 4.3.: Structured and unstructured information in a Wikipedia article about *"Michael Jackson, the English writer and journalist"*, in a revision from June 30, 2014.

Information about the nationality and profession can be listed in several places in a Wikipedia article, either visible in the article or invisible in form of metadata. Figure 4.3 shows a screenshot of the revision of a Wikipedia article about *"Michael Jackson, the English writer and journalist"*[47] which shows visible information. An article consists of unstructured text, seen on the left side of Figure 4.3. Some articles have info boxes on the right side, which provide structured data.

The first section is an introduction to the article which summarizes the most important points. Nationality and profession of a person are usually listed in the first sentence of this section (e.g., *"Michael Jackson (27 March 1942 - 30 August 2007) was an English writer and journalist."*). Since the article itself consists of unstructured data, it is not straightforward to extract all necessary data from the first sentence, especially because they are structured differently in each article.

Info boxes, on the other hand, provide structured information which can be extracted

[47]https://en.wikipedia.org/w/index.php?title=Michael_Jackson_(writer)&oldid=614982531

easily. However, there are several problems and inconsistencies with info boxes. First, there are articles that do not contain info boxes, especially articles about less known people. Second, some info boxes do not provide much information, even if the information is listed in the article. Third, there is no clear structure for the content of the boxes. Even some fields which refer to the same information can have different names. For instance, the field that lists a person's professions has several names, for example, `Occupation, Profession, Known for, Position, Playing position`.

Each Wikipedia article is assigned to at least one category, shown at the end of the article. Figure 4.4 shows categories for the article in Figure 4.3, as taken from the edit page including markup. The number of categories depends on each article. As with the info boxes there are no clear standards and the types of categories can be very different.

```
[[Category:1942 births]]
[[Category:2007 deaths]]
[[Category:Beer writers]]
[[Category:British columnists]]
[[Category:British Jewish writers]]
[[Category:English food writers]]
[[Category:English journalists]]
[[Category:Deaths from diabetes]]
[[Category:Deaths from myocardial infarction]]
[[Category:Deaths from neurological disease]]
[[Category:Deaths from Parkinson's disease]]
[[Category:English Jews]]
[[Category:Lithuanian Jews]]
[[Category:People educated at King James's Grammar School (Almondbury)]]
[[Category:People from Wetherby]]
```

Figure 4.4.: Examples of categories in Wikipedia, for *"Michael Jackson (writer)"*.

To make the process of obtaining articles about people with specific properties easier and clearer, we decided to use information from `Persondata`[48], a special type of metadata used in Wikipedia until May 2015. Persondata was a template added to biographical articles and provided structured information (e.g., name, alternative names, short description, date of birth, place of birth, date of death, and place of death). Since Persondata was not always kept up to date by editors, it was removed from articles in May 2015. The information stored in Persondata has been migrated to Wikipedia's sister project Wikidata[49].

The advantage of Persondata is that the parameter `SHORT DESCRIPTION` provides a concise description of a person including the nationality and profession (if known), for

[48]`https://en.wikipedia.org/wiki/Wikipedia:Persondata`
[49]`https://en.wikipedia.org/wiki/Wikidata`

```
{{Persondata
| NAME               = Jackson, Michael
| ALTERNATIVE NAMES =
| SHORT DESCRIPTION = British writer
| DATE OF BIRTH      = 27 March 1942
| PLACE OF BIRTH     = Wetherby, West
                       Yorkshire
| DATE OF DEATH      = 30 August 2007
| PLACE OF DEATH     = London
}}
```

(a) Michael Jackson, the British writer

```
{{Persondata
| NAME               = Jackson, Michael
| ALTERNATIVE NAMES =
| SHORT DESCRIPTION = Canadian actor
| DATE OF BIRTH      = November 8, 1970
| PLACE OF BIRTH     = Ottawa, Ontario
| DATE OF DEATH      =
| PLACE OF DEATH     =
}}
```

(b) Michael Jackson, the Canadian actor

Figure 4.5.: Examples of the Persondata field in Wikipedia.

example, *"American singer", "British writer", or "Canadian actor"*. This parameter is also filled even if there is no info box in the Wikipedia article. Examples of Persondata can be seen in Figure 4.5. We used this description to extract articles about people with certain properties. For example, to obtain articles with the property *"singer"* we extracted all articles that contained the word *"singer"* in the Persondata field.

For each entity in our pilot study, we created one properties corpus that consists of extracted documents that belong to the respective properties (nationality and profession). For example, to create the properties corpus for an American singer we randomly selected n documents about people who are American and n documents about people who are singers, and concatenated them into one "American singer corpus". For our experiments we chose $n = 500$. Some properties are rare in Wikipedia and we extracted less than 500 articles. In these cases we extracted additional articles with very similar, hand-selected properties[50]. For example, for *"gunfighter"* we extracted additional articles with the property *"gunslinger"* and for *"radio commentator"* we extracted additional articles with *"radio host"*. We then concatenated all articles for one property into the respective properties corpus.

For the experiments in our pilot study we created a limited set of properties corpora. We used a total of 15 nationalities and 83 professions which can be seen in Table 4.4. Choosing which nationalities and professions to use was done manually and with reference to the entities in our test set. Extracting and concatenating documents into the properties corpora was done automatically. In a setup going beyond our initial implementation, more nationalities and professions need to be extracted as it is not known which entities the system will be applied to. This can be done automatically, for example, by using lists of nationalities and professions.

[50]In a more advanced system, similar properties can be defined automatically, for example, by taking synonyms or semantically very similar words, for example, with the help of a lexical database such as WordNet (Miller, 1995).

Nationalities		
American	German	South African
Australian	Irish	Swiss
British	New Zealand	Tasmanian
Canadian	Niuean	Trinidadian
English	Scottish	Welsh

Professions		
academic	composer	musician
actor	convict	navy sailor
admiral	cricketer	offensive lineman
American football player	darts player	philosopher
anthropologist	director	poet
archer	engraver	politician
architect	equestrian	priest
army officer	explorer	professor
artist	farmer	racing driver
astronaut	figure skater	radio commentator
astronomer	film director	rower
athlete	footballer	rugby league player
Australian rules footballer	guitarist	rugby union player
baron	gunfighter	satirist
baseball coach	herbalist	sergeant
baseball player	hockey player	singer
baseketball player	hurler	skier
bassist	icehockey player	snooker referee
beachvolleyball player	industrialist	soccer player
bicycle racer	journalist	songwriter
bishop	judge	tennis player
botanist	lawyer	TV excecutive
boxer	leader	TV producer
brewer	linebacker	widereceiver
builder	martial artist	wine maker
colonial president	minister	wrestler
comedian	missionary	writer
comic editor	mormon leader	

Table 4.4.: Nationalities and professions used for properties corpora.

Silver standard test corpus

To evaluate our system we focus on a select number of people with the same name, aiming for high ambiguity within the dataset. Collecting and manually annotating such data is expensive, which is why we automatically created a dataset by exploiting the link structure in Wikipedia.

Such datasets, often called *silver standard*, have been created and used for NLP tasks before. Some of these datasets have been created by combining and harmonizing the annotation output from several independent systems (e.g., Moldovan and Novischi (2004); Rebholz-Schuhmann et al. (2010a,b); Hahn et al. (2010); Kang et al. (2012)). More recently, the link structure and metadata provided by Wikipedia has been used to automatically create datasets (e.g., Nothman et al. (2008); Nemeskey and Simon (2012); Nothman et al. (2013); Hahm et al. (2014)).

Wikipedia articles contain not only external links but also internal links to other Wikipedia articles. Consider the following sentence, including link markup, taken from the article *"Archbishop of Dublin"*[51]:

(4.2) *"The current [[Church of Ireland]] archbishop is the Most Reverend [[Michael Jackson (bishop)|Michael Jackson]], Archbishop of the [[Diocese of Dublin and Glendalough]]".*

This sentence contains three links to other Wikipedia articles. Links to internal Wikipedia articles are formatted as `[[article title]]` or `[[article title|link text]]`. In case there is no link text specified, the article title is used as the link text. These links are manually set by human editors whenever they feel that reference to another article is needed or useful. The correctness of these links are also checked by many other human editors, which makes them very reliable.

In addition to links to the articles *"Church of Ireland"* and *"Diocese of Dublin and Glendalough"*, the sentence contains a link to the article *"Michael Jackson (bishop)"*. If a person (or a term in general) is ambiguous in Wikipedia, these links always link to the correct corresponding entity because the human editors have already disambiguated it. This means we can use this type of link information as gold labels.

We chose 14 people with ambiguous names, for example, *"Michael Jackson"*, *"Paul Williams"*, and *"John Smith"*. We then searched all Wikipedia articles and each time we found one of these names linked to another article, we extracted the name (link text if specified, otherwise the link to the article), the actual link to the other article, and the context around the name. We experimented with three different context sizes:

[51] `https://en.wikipedia.org/wiki/Archbishop_of_Dublin`

- the sentence around the entity
- the paragraph around the entity
- the section around the entity

This means that for every occurrence of a name we found in a document, we extracted three text snippets, depending on the context. Table 4.5 shows example text snippets for all three context sizes extracted for the entity *"Michael Jackson (bishop)"*.

Context	**Text snippet**
Sentence	The current Church of Ireland archbishop is the Most Reverend **Michael Jackson**, Archbishop of the Diocese of Dublin and Glendalough.
Paragraph	From 1846 to 1977, Church of Ireland diocese of Dublin and Glendalough was united with the see of Kildare. The current Church of Ireland archbishop is the Most Reverend **Michael Jackson**, Archbishop of the Diocese of Dublin and Glendalough.
Section	The diocese of Dublin was formally established by Sigtrygg (Sitric) Silkbeard, King of Dublin in 1028, and the first bishop, Dúnán, was consecrated in about the same year. The diocese of Dublin was subject to the Province of Canterbury until 1152. At the Synod of Kells, held in March 1152, Dublin was raised to an ecclesiastical province with the archbishop of Dublin having the jurisdiction over the bishops of Ferns, Glendalough, Kildare, Leighlin and Ossory. In 1214, the dioceses of Dublin and Glendalough were united, which was confirmed by Pope Innocent III on 25 February 1216 and by Pope Honorius III on 6 October 1216. After the Reformation, there are apostolic successions of Church of Ireland and Roman Catholic archbishops. From 1846 to 1977, Church of Ireland diocese of Dublin and Glendalough was united with the see of Kildare. The current Church of Ireland archbishop is the Most Reverend **Michael Jackson**, Archbishop of the Diocese of Dublin and Glendalough. Sometime after the Reformation, Glendalough was dropped from the Roman Catholic archdiocese title. The current Roman Catholic archbishop is the Most Reverend Diarmuid Martin, Archbishop of the Archdiocese of Dublin, who succeeded to the title on 3 May 2003 and installed at St Mary's Pro-Cathedral, Dublin on 30 August 2003.

Table 4.5.: Text snippets for different context sizes.

Table 4.6 shows some general statistics about the extracted snippets for each proper name. More detailed statistics can be seen in Appendix B in Table B.1, which lists all real world entities and how many snippets exactly were extracted for each proper name.

The second column (Ent) in Table 4.6 lists the number of different real world entities for each proper name found in Wikipedia. The number of different entities is lower for some names (e.g., two different entities for *"Peter Müller"*, three for *"Roger Taylor"*, and four for *"Richard Burton"*). There are two names with a large amount of different entities, *"John Williams"* with 29 entities and *"John Smith"* with 38 entities. The average

Proper name	Ent	All	Max	%Max	Rest	%Rest	Avg(A)	Avg(R)
David Mitchell	9	201	111	55.22	90	44.78	22.33	11.25
David Thomas	9	85	47	55.29	38	44.71	9.44	4.75
Jack Johnson	8	427	230	53.86	197	46.14	53.38	28.14
John Edwards	6	418	400	95.69	18	4.31	59.71	3.00
John Smith	38	466	107	22.96	359	77.04	12.11	9.45
John Williams	29	852	650	76.29	202	23.71	28.37	6.97
Michael Collins	7	443	364	82.17	79	17.83	63.29	13.17
Michael Jackson	12	3968	3899	98.26	69	1.74	330.67	6.27
Michael Moore	7	566	532	93.99	34	6.01	62.89	4.25
Paul Williams	13	385	149	38.70	236	61.30	29.62	19.67
Peter Müller	2	15	10	66.67	5	33.33	7.50	5.00
Richard Burton	4	606	592	97.69	14	2.31	151.50	4.67
Roger Taylor	3	79	39	49.36	40	50.64	26.33	20.00
Tony Martin	13	275	91	33.09	184	66.91	21.15	15.33
Average	11.43	627.57	515.79	65.66	111.79	34.34	62.75	10.85

Table 4.6.: Statistics for extracted snippets for each name. A=All, R=Rest. Numbers are counts or percentages.

number of real world entities per name is 11.43.

The third column (All) in Table 4.6 shows the overall numbers of context snippets we extracted for each name, summing up the snippets we extracted for each entity with the same name. Some people are more well-known than others (e.g., *"Michael Jackson, the American singer"* or *"John Williams, the American composer"*), resulting in obtaining a larger number of snippets. The average number of extracted snippets is 627.57. This average number is so high because we extracted a vast amount of snippets for *"Michael Jackson"*. Other than that and *"John Williams"*, all other names have less than the average number of snippets.

The fourth column (Max) shows the maximum numbers of snippets for a single entity with that name. For example, we extracted a total of 3,968 snippets for the name *"Michael Jackson"* (All). Out of these snippets, 3,899 snippets (Max) belong to the entity *"Michael Jackson, the American singer"*. The remaining 69 snippets belong to the other eleven entities with the same name (see Table B.1 for details on these other entities). For each name, the entity with the most snippets is used for the majority class baseline for the respective name. The average maximum number of snippets is 515.79. As with the overall number of snippets, this high average number is due to the entity *"Michael Jackson, the American singer"*.

There are some extreme cases like the one with *"Michael Jackson"* where one entity has many more snippets than the others because it concerns a very famous person. The fifth column (%Max) gives the percentage of snippets which belong to the majority class for the name, i.e., to the one entity with most snippets. There are four persons which take up over 90% of the snippets extracted for the respective names (*"John Edwards", "Michael Jackson", "Michael Moore", "Richard Burton"*). In other cases, the number of snippets is more evenly distributed over all entities with the same name. On average, the majority class consists of 65.66% of the snippets.

In columns six (Rest) and seven (%Rest) we list the number of extracted snippets and the respective percentages for the remaining entities, i.e., the entities that do not belong to the majority class. The average number of snippets for the remaining entities is 111.79 (34.34%).

Column eight (Avg(A)) lists the average numbers of snippets that were extracted for each name. The average number is especially large for the names that have a very large majority class. On average, the number of snippets is 62.75. To gain a better understanding of the remaining snippets, we list the average number of snippets excluding the majority class in the last column (Avg(R)). For most names, the average number of snippets for each of their entities is less than 10. For a few of them, it ranges between 11.25 and 28.14. The average over all names minus the majority class is 10.85 snippets.

Preprocessing

We used the Java Wikipdia Library (JWPL) (Zesch et al., 2008) to extract documents from Wikipedia. JWPL gives access to information in Wikipedia including metadata (e.g., titles, section information, category information, Persondata). We used some of this metadata to extract documents for our different corpora. To extract documents for the copora used to train the topic models we used category information to create W_{lp} (category *living people*), and category and section information to create W_{lps} (category *living people* as well as information about individual sections). For the properties corpora we used the parameter `SHORT DESCRIPTION` in the Persondata metadata to extract documents about people with specific properties. After extracting the articles, Wikipedia markup and images were filtered out to obtain plain text only.

For the English Gigaword corpus we used a subset of the corpus (the *New York Times Newswire Service* section) as described in Section 4.3.1. We extracted this subset by taking all files that contained the abbreviation of the section (NYT) in the file name. The files contain plain text without metadata, therefore no further preprocessing was necessary.

To obtain topic information we used the MALLET toolkit (McCallum, 2002), which contains several machine learning tools, for example, document classification, clustering, topic modeling, and information extraction. For topic modeling it provides implementations of latent Dirichlet allocation (LDA), pachinko allocation, and hierarchical LDA. We used the `ParallelTopicModel` class which is a simple parallel threaded implementation of LDA based on Newman et al. (2009). It utilizes a SparseLDA sampling scheme and data structure from Yao et al. (2009).

4.3.2. Experimental setup

Parameters for topic models

We first trained different topic models using the corpora and number of topics described in Section 4.3.1[52]. We then applied the trained topic models to the properties corpora as well as the context snippets of our silver standard corpus to obtain topic information from them. The results for each corpora and context snippet is a list of topics with the corresponding probabilities of the topics with respect to the corpus or snippet. We noticed that often only 1-3 topics have a high probability while most topics have a probability below 1%. This shows that our approach does learn the most important topics from the documents.

Since topics with a low probability are not very reliable in identifying a person, we only took the 10 topics with the highest probability for further calculations for both the properties corpora and the context snippets. We then compared the vector representing the topic information from each context snippet (i.e., each new entity) with each vector consisting of topic information from the properties corpora as described in Section 4.2.2.

Baselines

We use two baselines for our experiments.

1. **Baseline Majority (BM):** All labels in the data are assigned to the majority class.

2. **Baseline Jaccard (BJ):** Uses the Jaccard index to compare the similarity between the context snippet of the new unknown entity and the properties corpora we created. The Jaccard index is defined as the size of the intersection divided by the size of the union of two sample sets A and B:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{4.2}$$

[52]The other parameters were: $\alpha_t = 0.01$, $\beta_w = 0.01$, number of iterations (training the model) = 1000, number of iterations (inducing new documents) = 10, thinning = 1, burn-in = 5, with α_t being the sum over topics and β_w the parameter for a single dimension of the Dirichlet prior.

4.3.3. Results and discussion

We conducted a total of 546 experiments[53] with the following parameters:

- 14 different names (see Section 4.3.1, Table 4.6)
- 3 different context sizes (sentence, paragraph, section)
- 6 different number of topics (100, 1,000, 2,500, 5,000, 7,500, 10,000)
- 4 different corpora to train the topic model (3× Wikipedia, 1× English Gigaword)

In this section we discuss the results of these experiments. However, due to the large number of results we do not show them all. We first illustrate the general tendency of the results by using one setting of parameters. We chose parameters which had an average performance in the experiments to give an estimate of how the approach works. We also compare and discuss the effect of different parameters.

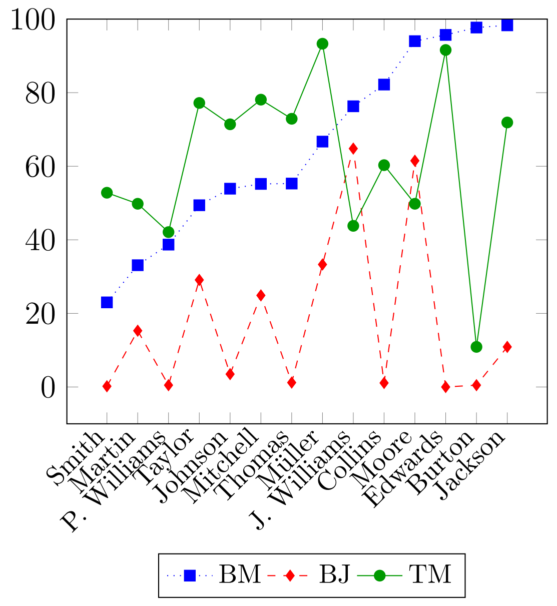
Micro-averaged F_1 score and macro-averaged F_1 score

Figure 4.6 shows the F_1 scores for all experiments with the following parameters: context size=paragraphs, topics=1000, corpus=W_{all}. The names are ordered by the micro-averaged F_1 score of the Baseline Majority (BM) in ascending order, i.e., from worst to best results. In all subsequent graphs we use the same ordering for better comparison.

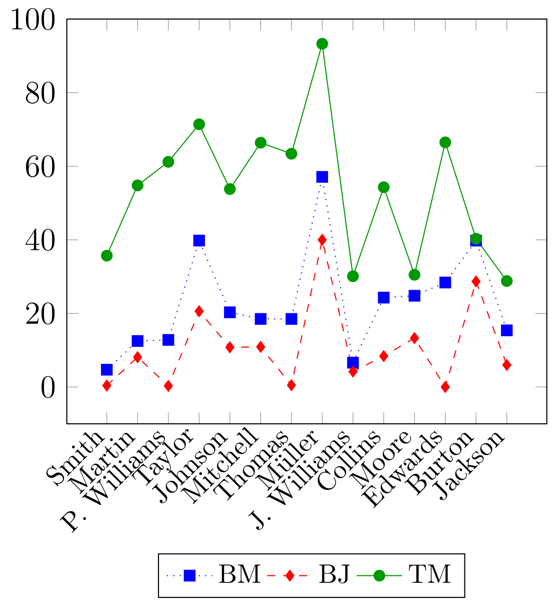
We present both the micro-averaged F_1 score (Figure 4.6a) and the macro-averaged F_1 score (Figure 4.6b), which give quite different results. This is because both measures aim at different goals with how they weight things. Micro-averaged F_1 score weights each classification decision equally, i.e., it favors large classes, while macro-averaged F_1 score weights each class equally, i.e., it shows the effectiveness of small classes better (Manning et al., 2008). For a more detailed explanation and an example we refer to Chapter 2, Section 2.1.4.

Figure 4.6a shows that when using micro-averaging, in some cases the majority baseline (BM) is better than our topic model approach (TM). This is the case when there is one famous name-bearer who has many more snippets than the other persons with the same name (e.g., *"Moore", "Edwards", "Burton", "Jackson"*), which results in one class being much larger than the others (for more details on the class distribution of these and other cases see Table B.1 in Appendix B). For example, our test set contains 3,968 snippets for *"Michael Jackson"*, with 3,899 of the snippets belonging to the majority class (*"Michael Jackson, the American singer"*), which results in a micro-averaged F_1 score

[53]Since we limited the number of topics for some corpora, we do not obtain $6 \times 4 = 24$ different settings for topic number + corpora, but $2 + 6 + 2 + 3 = 13$ (see Table 4.3 for details). This leads to $14 \times 3 \times 13 = 546$ experiments.



(a) micro-averaged F_1 score



(b) macro-averaged F_1 score

Figure 4.6.: F_1 score results of Baseline Majority (BM), Baseline Jaccard (BJ), and Topic Modeling (TM) approaches with parameters: context size=paragraphs, topics=1000, corpus=W_{all}.

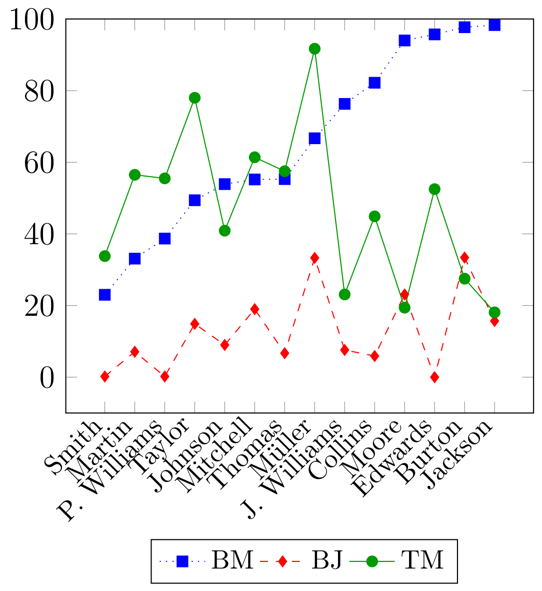
of 98.26% for BM. This is expected behavior as micro-averaging favors large classes as we pointed out before. In cases with more evenly distributed classes our TM approach generally performs better than BM.

The Baseline Jaccard (BJ) stays even below the Baseline Majority in most cases. It generally only works for entities which have many snippets in our test set. The reason is that these entities are more well-known and the contexts we extracted for these entities were often longer and provided more information which is needed to obtain a suffiently sized match with the Jaccard algorithm. Smaller classes, on the other hand, often had small contexts and did not provide as much information as is needed for this baseline approach.

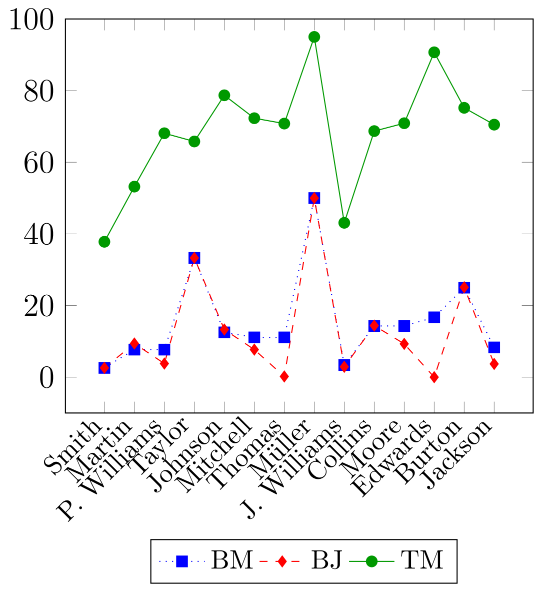
The macro-averaged results in Figure 4.6b give a better sense of how well our approach works on smaller classes. Our approach performs better than both baselines in all cases, including the cases where most entities are in the majority class. The entities in our test set serve as the "little-known" referents since we left out explicit material about them from the training data. However, even within these test entities there are some that are more well-known (mostly the people that end up in the majority classes), while many other people are less known (small classes). Our approach works well for these less known entities, which confirms that the approach works well on little-known entities without training material.

Macro-averaged precision and recall

In Figure 4.7 we give more insight into the macro-averaged results of Figure 4.6b and show the macro-averaged precision (Figure 4.7a) and macro-averaged recall (Figure 4.7b) with the same parameters as before (context size=paragraphs, topics=1000, corpus=W$_{all}$).



(a) macro-averaged precision



(b) macro-averaged recall

Figure 4.7.: Precision and recall results of Baseline Majority (BM), Baseline Jaccard (BJ), and Topic Modeling (TM) approaches with parameters: context size=paragraphs, topics=1000, corpus=W$_{all}$.

Precision-wise, the Baseline Majority does better in cases that have one large class. This is expected because if most text snippets in the test set belong to the majority class, the number of false positives is small, which leads to higher precision. Our TM approach beats BM in cases where the sizes of each class are more evenly distributed.

Figure 4.7b shows that the recall for our approach outperforms both baselines by far in all cases. In one case (*"Edwards"*) we even achieve a recall of 98.6%. The baseline approaches do not work well for the generally less known people with small classes in our test set. BM does not work well because it incorrectly tags the small classes with the majority class and BJ does not work well because the information in the context snippets is not sufficient for this approach. Measuring the similarity between words, n-grams, or similar approaches suffers from sparsity problems. Our TM approach makes better use of smaller context snippets because it extracts and uses relevant information (i.e., the topic information) about a person more effectively. This leads to higher recall results in most cases. In some cases with different parameters, we even achieve a recall of 100% for some entities (see Table 4.7 and Table 4.8). Obtaining a better recall is important in applications that aim for high recall results (e.g., in search engines or QA systems). With a high recall a system returns more correct results instead of returning results of similar persons which are more well-known. Our TM approach is therefore well-suited for systems which have the goal of returning high recall results.

In Table 4.7 we show details for one of the proper names in the previous results, *"David Mitchell"*, and the nine real world entities it can refer to in our test set. The parameters are the same as in the previous figures (context size = paragraphs, topics = 1000, corpus = W_{all}). The number of context snippets we extracted for each real world entity of the name is listed in the second column (#Ex), ranging from 2 to 111 with a total of 201 snippets. The rest of the table shows precision (P), recall (R) and F_1 score for the two baselines BM and BJ as well as our topic model approach TM.

The majority baseline (BM) only labels the majority class of the snippets correctly (in this case *"comedian"*) which gives a micro-averaged F_1 score of 55.2%. In cases where one name refers to one very well-known person (e.g., *"Michael Jackson"*) the micro-averaged F_1 score is much higher because almost all snippets belong to the majority class. In this case the difference (in terms of class size) between the majority class and other classes is not very large which results in a lower micro-averaged F_1 score for this name.

The Jaccard baseline (BJ) only works well for the two most well-known entities here (*"comedian"* and *"author"*). It generally does not work well for less known persons. As mentioned before, the reason is that the contexts extracted for these entities are often smaller and do not provide as much information as is needed for this baseline approach. The micro-averaged F_1 score for BJ is 24.9%.

Our topic model approach (TM) outperforms both baselines in almost all cases of precision, recall, and F_1 score with two exceptions. First, the BM recall performs better for the majority class (*"comedian"*) because it simply labels all text snippets with the majority class and therefore predicts all instances belonging to the majority class correctly, resulting in a perfect recall. Second, the BJ recall is better than the other approaches for the class *"author"*. As it can be seen in Table 4.7, the TM approach works well for more than half of the classes, with the F_1 score ranging from 58.4% to 100%. In a few cases (*"politician", "builder", "field hockey player"*) the results stay below 20%. In one case (*"lawyer"*) there were no true positives in the classification results which leads to a precision, recall and F_1 score of zero. The overall micro-averaged F_1 score is 68.2% and the overall macro-averaged F_1 is 52.8%.

As a comparison, Table 4.8 shows results of the same proper name with the same real world entities but with different parameters (context size = paragraphs, topics = 100, corpus = W_{lp}, i.e., less topics and using a topic model which learned from a more

		P			R			F_1		
Label	# Ex	BM	BJ	TM	BM	BJ	TM	BM	BJ	TM
comedian	111	55.2	35.5	**94.8**	**100.0**	19.8	82.0	71.2	25.4	**87.9**
author	57	0	21.4	**81.2**	0	**49.1**	45.6	0	29.8	**58.4**
fighter	11	0	0	**61.1**	0	0	**100.0**	0	0	**75.9**
politician	10	0	0	**11.1**	0	0	**20.0**	0	0	**14.3**
builder	3	0	0	**10.0**	0	0	**33.3**	0	0	**15.4**
field hockey player	3	0	0	**11.11**	0	0	**66.7**	0	0	**19.0**
Navy officer	2	0	0	**50.0**	0	0	**100.0**	0	0	**66.7**
figure skater	2	0	0	**100.0**	0	0	**100.0**	0	0	**100.0**
lawyer	2	0	0	0	0	0	0	0	0	0
all	201						micro-avg F_1:	55.2	24.9	**68.2**
							macro-avg F_1:	18.5	10.9	**52.8**

Table 4.7.: Results for the proper name *"David Mitchell"*, with the following parameters: topics=1000, corpus=W_{all}, context size=paragraphs. BM = Baseline Majority, BJ = Baseline Jaccard, TM = Topic Model Approach. Numbers are counts or percentages.

		P			R			F_1		
Label	# Ex	BM	BJ	TM	BM	BJ	TM	BM	BJ	TM
comedian	111	55.2	35.5	**90.7**	**100.0**	19.8	87.4	71.2	25.4	**89.0**
author	57	0	21.4	**90.0**	0	49.1	**63.2**	0	29.8	**74.2**
fighter	11	0	0	**100.0**	0	0	**100.0**	0	0	**100.0**
politician	10	0	0	**23.8**	0	0	**50.0**	0	0	**32.3**
builder	3	0	0	**23.1**	0	0	**100.0**	0	0	**37.5**
field hockey player	3	0	0	0	0	0	0	0	0	0
Navy officer	2	0	0	**100.0**	0	0	**100.0**	0	0	**100.0**
figure skater	2	0	0	**100.0**	0	0	**100.0**	0	0	**100.0**
lawyer	2	0	0	**25.0**	0	0	**50.0**	0	0	**33.3**
all	201						micro-avg F_1:	55.2	24.9	**78.1**
							macro-avg F_1:	18.5	10.9	**66.4**

Table 4.8.: Results for the proper name *"David Mitchell"*, with the following parameters: topics=100, corpus=W_{lp}, context size=paragraphs. BM = Baseline Majority, BJ = Baseline Jaccard, TM = Topic Model Approach. Numbers are counts or percentages.

specialized corpus). The results are similar to the ones in Table 4.7. The results for BM and BJ are the same because only the parameters concerning the topic modeling were changed, not the parameter for the text snippets. As before, the TM approach beats the baselines in all cases and improves recall for the class *"author"*. The BM results for the majority class are again better for the same reason discussed above.

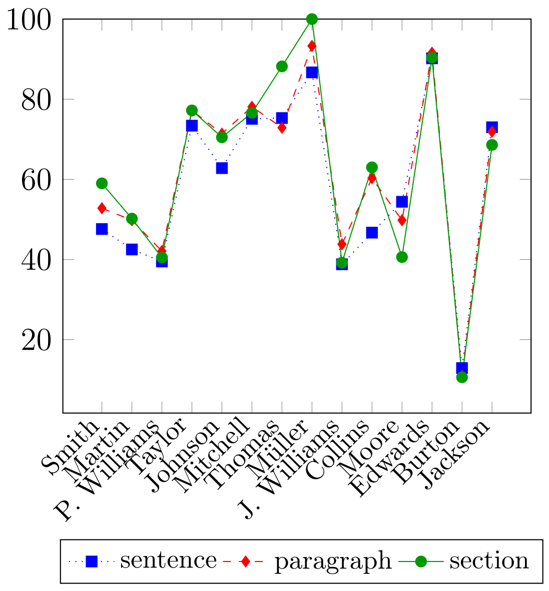
Our TM approach shows an an overall improvement compared to the results in Table 4.7, with a higher micro-averaged F_1 score of 78.1% and a higher macro-averaged F_1 score of 66.4%. The F_1 score of the TM approach improves in all cases, except for the entity *"field hockey player"*. In this case no true positives could be identified and all scores drop to zero. For this entity, using less topics and a more specialized corpus hurts the identification process. The other entities achieve F_1 scores between 32.3% and 100%. While in Table 4.7 only one entity had a perfect F_1 score of 100%, in Table 4.8 this applies to three entities. This shows that name disambiguation depends on the parameters used and by changing the parameters, better results can be achieved. We investigated the parameters further and discuss our findings in the next section. More details about the results discussed in Table 4.7 and Table 4.8 including the numbers of true positives, false positives, false negatives, and true negatives can be found in Appendix B in Table B.2 and Table B.3.

The results in this section show that the TM approach works not only well for identifying well-known persons but especially for identifying generally less known persons where often not much training material is available. The reason for this is because the topics are more general while measuring the similarity between words, n-grams, or similar approaches suffers from sparsity problems.

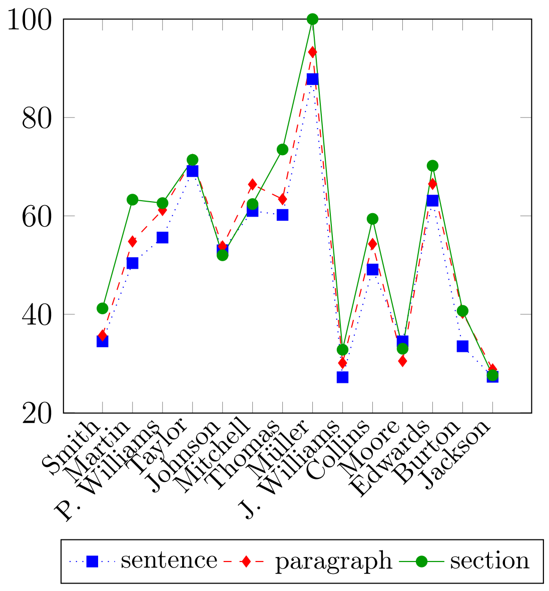
The TM approach does not beat the baselines for all names that we tested. In cases where the baselines perform better, there is usually a very well-known person in the set of people (with a very large number of context snippets compared to other people) that one or both baselines can easily identify. The main advantage of our TM approach is that it works well for unknown entities which usually have little or no available training material, and for entities which have little surrounding context.

Investigation of parameters

We investigated how the context size, number of topics, and the data used for training the topic model influence the results of our TM approach. For each of these three parameters we present a pair of graphs showing the micro-averaged F_1 score and the macro-averaged F_1 score. The parameter that is investigated is broken down into its possible values. The remaining two parameters have fixed values.



(a) micro-averaged F_1 score



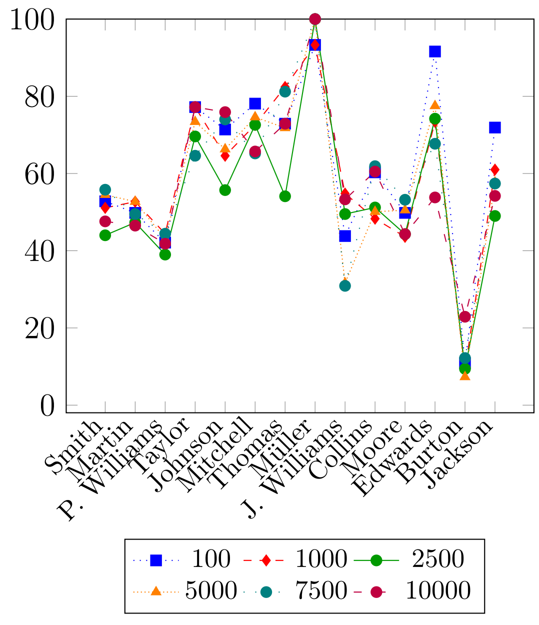
(b) macro-averaged F_1 score

Figure 4.8.: F_1 score results of different context sizes of TM approach with other fixed parameters: topics=1000, corpus=W_{all}.

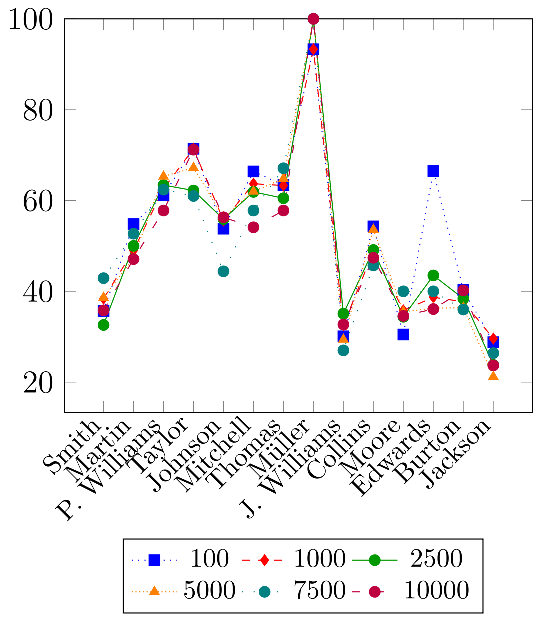
Figure 4.8 shows results for the different context sizes (sentence, paragraph, section), when using the same parameters for the topic model as in the previous figures (topics=1000, corpus=W_{all}). In most cases, taking more context gives better results (i.e., paragraphs are better than sentences and sections are better than paragraphs). This is expected to some extent because with a larger context the topic model can use more information to produce better results. In some cases, however, a larger context introduces more noise which leads to worse results than when taking a smaller context (e.g., *"Moore"* in Figure 4.8a).

Generally, taking a smaller context does not worsen the results much and in some cases gives nearly the same results as taking a larger context. This shows that our TM approach works well if there is only little context available for a person. This is important because the context for a person is often rather limited. For example, some documents are very short (e.g., short biographies). Longer documents are often about multiple persons and in some cases most of the document is not about the entity that needs to be disambiguated. In such cases, it is more useful to use a small context window or extract relevant context as we described in Chapter 3.

Table 4.9 shows results of our TM approach when using different numbers of topics for the topic model (100, 1,000, 2,500, 5,000, 7,500, 10,000). The fixed parameters are the same as in the general results we showed before (context size=paragraphs, corpus=W_{all}). There is no clear answer as to what is the best number of topics to train the topic model.



(a) micro-averaged F_1 score



(b) macro-averaged F_1 score

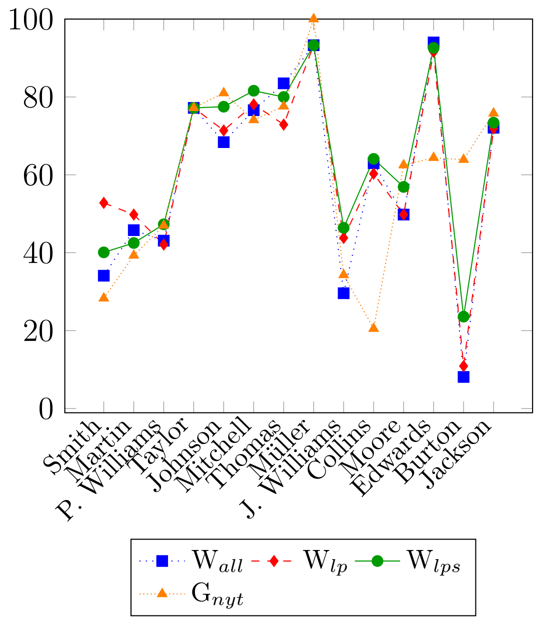
Figure 4.9.: F_1 score results of different number of topics using TM with other fixed parameters: context size=paragraphs, corpus=W_{all}.

In many cases, taking a small number of topics (like 100) seems to work best. Using a higher number results in more fine-grained topics. These topics are more specialized and can make finding the most similar properties corpus harder because while they both share very similar topics, they might not share the exact topics and the cosine similarity measure fails to identify the best corpus.

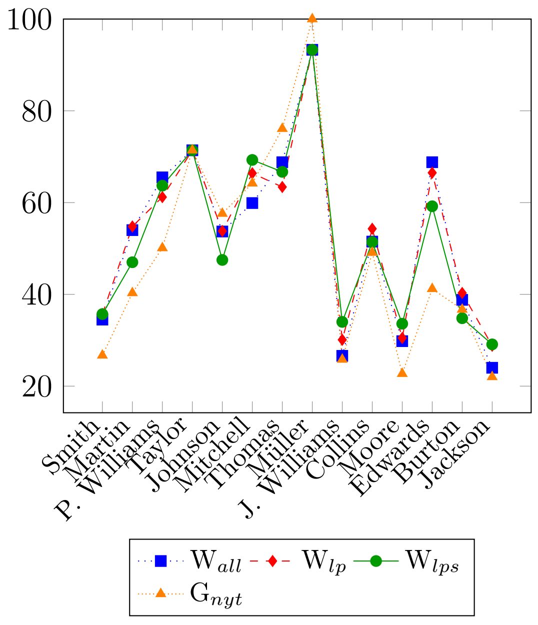
These results show that it is possible to take a small number of topics for our approach to achive good results. An advantage of using fewer topics is that it takes less time for the topic model to determine the topics of a new document. In a real application it is important to reduce the time needed as much as possible because a user of the application does not want to wait for several minutes or even hours for results.

In Table 4.10 we present results for different types of corpora used for training the topic model (W_{all}, W_{lp}, W_{lps}, G_{nyt}). The remaining parameters are fixed as in the previously shown general results (topics=1000, context size=paragraphs). There is no clear preference as to which corpus is the most useful one for the task and it also depends on the context size. Overall, using the entire Wikipedia seems to be the least helpful. This might be due to this corpora being too general which makes it harder for a topic model to learn proper topics. Evidently, it seems to be better to use a more specific collection which produces better topics for the task.

Moreover, we found that when using different collections, the results can change greatly. While the results for using different parts of the Wikipedia collection all fol-



(a) micro-averaged F_1 score



(b) macro-averaged F_1 score

Figure 4.10.: F_1 score results of different corpora for models using TM with other fixed parameters: topics=1000, context size=paragraphs

lowed the same trend, the results for using the Gigaword corpus were quite different and often worse than results using Wikipedia. This is probably due to the English Gigaword corpus being a newswire corpus and thus having less variety of topics than Wikipedia articles.

4.3.4. Error analysis

We performed an error analysis on the text snippets that were incorrectly classified. In this section we discuss problems that we found in this error analysis. For each example we provide the correct entity (`C`) and the incorrect entity (`I`) from the system output.

A general problem we noticed is that some of the text snippets do not consist of full sentences, but are only parts of lists as in examples 4.3a and 4.3b:

(4.3) a. *"Issue 37: Michael Jackson, The Jackson 5, Wah Wah Watson, DJ Nicky Siano."*

`(C: American singer | I: American radio commentator)`

b. *"13 – Michael Jackson First No.1: 'Ben' (1972)."*

`(C: American singer | I: British writer)`

Our test data consists of sentences, paragraphs, and sections extracted from Wikipedia articles with the Java Wikipedia Library. Some sections in Wikipedia consist of mostly or only lists (e.g., magazine issues or publication lists). These lists can be formated in different ways: usually they are formatted using HTML list markup (`<ul><li></li></ul>`), but in some cases Wikipedia editors use HTML paragraph markup (`<p></p>`) to format single list items. This results in each list item being treated as its own paragraph. List items generally do not contain full sentences but simply a few keywords as in examples 4.3a and 4.3b. This obviously makes it hard for our approach to identify the correct entity.

In some cases, the sections and paragraphs consist of exactly one sentence. This sentence can contain fewer useful information as in example 4.4a or more useful information as in example 4.4b:

(4.4) a. *"Her nickname is originated from a similarity with her idol, Michael Jackson."*
`(C: American singer | I: New Zealand anthropologist)`
b. *"University of California, Berkeley offers a course on racial transformation, including references to Michael Jackson."*
`(C: American singer | I: New Zealand anthropologist)`

If the sections and paragraphs are essentially the same as the sentence snippet, then the results for these sentence/section/paragraph snippets are also the same. There cannot be an improvement if the context does not change.

A general problem is that sentences, sections, and paragraphs often contain words that give clues about another person. Consider examples 4.5a and 4.5b:

(4.5) a. *"University of California, Berkeley offers a course on racial transformation, including references to Michael Jackson."*
`(C: American singer | I: New Zealand anthropologist)`
b. *"He also designed the Captain EO video starring Michael Jackson for Disney and the Steven Spielberg film Hook."*
`(C: American singer | I: British TV executive)`

In both examples, the correct entity is *"Michael Jackson, the American singer"*, but due to certain words in the context they were incorrectly identified as different people. In example 4.5a, words such as *"university"* and *"course"* lead to the result being *"the New Zealand anthropologist"*. In example 4.5b, the wrong result was *"the British TV executive"* because of words such as *"video"* and *"film"*. This problem can happen especially in short documents or when most of the document is about another person.

A similar problem is that sometimes there are no helpful clues in the text as in examples 4.6a–4.6c:

(4.6) a. *"At Branca's first wedding, Michael Jackson was Best Man and Rev."*
`(C: American singer | I: Canadian actor)`

b. *"Macdonald repeatedly ridiculed public figures such as Marion Barry, Michael Jackson and O. J. Simpson."*
`(C: American singer | I: American radio commentator)`

c. *"David Mitchell was the best man."*
`(C: British comedian | I: Australian builder)`

In our work we used nationalities and professions as properties and the sentences in examples 4.6a–4.6c do not contain words that are useful to identify the proper name through these characteristics. In such cases, a larger context is necessary. Other proper names in the context can be helpful to identify a person if they generally appear in the context of said person. However, it is hard to capture other persons with a topic model approach.

4.4. Related work

Related work on named entity disambiguation and especially on named entity linking typically focuses on a handcrafted set of well-known entities, or limits linking to entities available in a knowledge base (cf. our discussion of previous work on named entity linking and wikification in Chapter 2, Section 2.2). Techniques for dealing with entities which do not appear in the knowledge base are (i) ignoring the entity, (ii) mapping the entity to a special entity e_{out}, or (iii) returning a `NIL` result. Only little work has attempted to disambiguate little-known or out-of-knowledge-base referents and most systems focus on well-known entites for which datasets or other sources provide a large amount of training data (Ilievski et al., 2017). As we noted in Chapter 3, Section 3.3, many of the existing datasets used for named entity disambiguation do not contain a great amount of ambiguity. Little ambiguity and datasets being biased towards well-known entities was also found in other studies (e.g., van Erp et al. (2016); Ilievski et al. (2016)). This leads to what the authors of these studies call "semantic overfitting" and contributes to systems performing better on datasets containing little ambiguity and biased towards well-known entities (Ilievski et al., 2018).

Even though most NED systems still focus on well-known entities, the need to disambiguate little-known referents has become more recognized by researchers in recent years. This can be partly attributed to the growth of social media where one can find a plethora of little-known entities and more researchers investigating the diambiguation of entities in social media and other web sources. One research area of recent years focuses on *emerging entity discovery*, also called *emerging entity detection* (e.g., Hoffart et al. (2014); Färber et al. (2016); Hoffart et al. (2016); Zhang et al. (2019); Akasaki et al. (2019)). An emerging entity is usually defined as an entity that is just about to start to become popular in the future. Such entities are not added immediately to knowledge bases like Wikipedia, but only after they attracted enough public interest which can take a while depending on the entity (Keegan et al., 2013; Fetahu et al., 2015). Since emerging entities are by definition in the process of becoming more well-known, there is generally a good amount of textual material about them available already. This means that while this research area deals with out-of-knowledge-base entities, it does not disambiguate little-known entities which do not become popular and for which only little material is available.

Little-known referents are sometimes referred to as *long tail entities* in the literature. In a probability distribution, the *head* refers to a small set of values with a high frequency, while the *tail* refers to the large (or "long") set of values with a low frequency. Ilievski et al. (2018) performed a systematic study on the long tail in the context of the entity linking task and found that the distribution of entity mentions in text has a Zipfian distribution (Zipf, 1949), i.e., there is a long tail of entities which appear with a low frequency. However, the term has different meanings depending on how the authors define it in the context of their systems. Especially in the context of entity linking to a knowledge base, what authors mean with "long tailed" entities are less known entities in the knowledge base, not less known entities in general. Examples of such work include Hoffart et al. (2012), who propose the system KORE that incorporates weighted (multi-word) keyphrases that can partially overlap with other phrases to find the best candidate (e.g., *"English rock guitarist"* is more similar to *"English guitarist"* than to *"German president"*). Nguyen et al. (2014) present AIDA-*light* and use Wikipedia categories (e.g., `wiki-category:English_footballers`) to obtain further information in cases of poor context information. Bhagavatula et al. (2015) weaken the usually strict assumption that an entity is to be linked to pre-defined types and relations. Their system *TabEL* gathers sets of coherent entity pairs that tend to co-occur in Wikipedia documents and tables, and uses soft constraints on them. All these approaches are restricted to linking entities to the existing entries in the knowledge base and cannot disambiguate out-of-knowledge-base entities.

Some approaches have recognized the importance of not ignoring entities which are not present in knowledge bases and attempt to resolve the problem by assigning them a fine-grained semantic class (e.g., *"musician"* or *"album"*). This method is adopted from the task known as *(fine-grained) semantic typing or classification of entities* (e.g., Fleischman and Hovy (2002); Rahman and Ng (2010); Ling and Weld (2012); Yosef et al.

(2012); Yogatama et al. (2015)), which assigns fine-grained types to named entities as opposed to coarse-grained types typically used in named entity recognition. Two approaches that assign fine-grained types to out-of-knowledge-base entities are (Lin et al., 2012) and (Nakashole et al., 2013).

Lin et al. (2012) address entity linking at web scale and attempt to link the subjects of 15 million entity-based assertions, which were extracted from 500 million web documents, to their corresponding Wikipedia articles. For example, the assertion *"New York", "acquired", "Pineda"* would be linked to the articles *"New York Yankees"*. They found that out of these 15 million assertions, around 5 million assertions (33.3%) could not be linked to an article. This was often the case for very new or non-prominent entities (e.g., *"fiscal year 2012"*), as well as non-entities (e.g., *"serious change"*). If entities cannot be linked to an article, they assign a semantic category to the entity by propagating the semantic category from similar entities found via relations (e.g., *"X is a good source of Y"* or *"X teaches at Y"*). Nakashole et al. (2013) focus on out-of-knowledge-base noun phrases and assign fine-grained semantic classes to them. In their work they use relational patterns (e.g., *"X released Y"*) to obtain possible types for the entities, and a co-occurrence likelihood weight computation to give weights to the candidate types. They include type disjointness contraints to filter out disjoint candidate-candidate pairs (e.g., `person-artifact`) and type correlations to give higher probability to combinations that are strongly correlated (e.g., *"guitar player"* and *"electric guitar player"*).

While these approaches return at least some information instead of ignoring the entity, they are restricted to assigning a semantic class that provides limited information (e.g., profession). This can help disambiguate names when the entities are very different. However, these approaches might not be able to disambiguate, for example, different singers or actors that share the same name. Our approach includes more properties in the pilot study (profession and nationality) and is more flexibel in general in that it can incorporate further properties.

To overcome the problem of semantic overfitting on datasets, Postma et al. (2016) propose requirements for designing new datasets as well as an example task for event-based question answering that should not only deliver reliable results of well-known entities but also on the long tail of entities. To this end they define several requirements for such a system and explicitly include a requirement that covers little-known entities with little presence in media. However, the authors do not implement their requirements and to this date no one has taken up on the proposed requirements. Esquivel et al. (2017) study long tail entities in news corpora and map them to semantic classes. Their findings reveal that up to 50% of the long tail entities found in news cannot be linked to a Wikipedia article. They do not attempt to disambiguate the entities. Garigliotti et al. (2019) are concerned with extracting contexts for entities and specifically focus on long tail entities. For their test set they collect 165 long tail entities (of which 3 are people and the rest are organizations) by hand and search for contexts that contain the entity at least once. They show that their unsupervised approach can handle retrieving contexts for long tail entities well.

While most NED/EL systems still focus on identifying well-known entities, it has been shown in the research discussed in the last few paragraphs that little-known referents are important (both for evaluating a system without bias towards well-known entities and to obtain more reliable results in applications such as QA systems). News and social media generate texts with entities that are just emerging or might never be popular enough to be added to a knowledge base. Applications that want to provide reliable results need to take these entities into account and provide robust methods to identify them. This is where our approach is a good base for identifying little-known referents.

Topic models have been used for named entity disambiguation in related work. Bhattacharya and Getoor (2006) examine data with authors and try to determine the number of individual authors and link them to their corresponding entities. They do not make pairwise decisions of whether two names refer to the same entity, but detect all names in the collection and make a collective decision. For this they use a latent Dirichlet allocation (LDA) model and introduce a hidden variable that captures relationships between entities. With this approach they extend work done by Rosen-Zvi et al. (2004) and propose a solution to the duplicate authors problem. Shu et al. (2009) extend the LDA model to include global information from documents to identify authors. All of these approaches use author data only. By contrast, we propose an approach to disambiguate people in general.

Some work has been done using topic models on knowledge bases. Zhang et al. (2011) train a topic model on a knowledge base, then incorporate the information as a semantic feature by mapping documents with the name mention to the hidden topic space. They use Wikipedia pages to train their model on the first part of the pages and to test it on the second part of the pages. Sen (2012) use topic models to learn context information and relations between co-occurring entities. They train their model on only those Wikipedia articles which describe the entities they are dealing with. Still, to use their model for other entities they would have to use an expanded version of the knowledge base. This would require the knowledge base to contain information about these entities. Han and Sun (2012) combine context compatibility of a referent entity and its context as well as topic coherence of the target entity and the document's main topics. Kataria et al. (2011) use a hierarchical variant of LDA that incorporates information from Wikipedia (words, annotations, and category information) and have a separate topic for each Wikipedia entry in their model. They report results on precision whereas we also consider the recall and F_1 score, and compare different parameters.

All of these approaches are limited to the entities that occur in the knowledge bases. To apply the approaches to other entities, the knowledge base would have to be extended or data about these entities would have to be collected otherwise. Despite the fact that our approach also uses Wikipedia as a collection to train some of our topic models, the information we obtain is independent from specific entities and can be applied to any new entity, even if there is no information about them available in Wikipedia.

Li et al. (2013) use information from Wikipedia and an external source (websites referring to entities in Wikipedia obtained by crawling the web). They conduct experiments on two datasets, TAC-KBP 2009 and twitter data about 25 ambiguous and randomly picked entities. However, they filter this data to only include entities that occur in Wikipedia, because their approach is also limited to entities in Wikipedia.

Bamman et al. (2013) extend a topic model to learn character types of people in movies (e.g., `{dark, major, henchman}` or `{shoot, aim, overpower}`). In subsequent work, Bamman et al. (2014) apply an extended topic model to learn character types in English novels of the 18th and 19th century. However, they do not identify and link the individual names to their real world entities.

4.5. Summary

In this chapter we introduced a new system for named entity disambiguation which does not need specific textual training data about a person to disambiguate them. Our approach works in two steps. First, we learn properties (e.g., nationalities and professions) from documents about people sharing the same properties by using a topic model approach. Second, we apply this topic information on text snippets of a new unseen person by obtaining the topic information of the person and then comparing it to the information we collected earlier to determine which properties are closest to the person.

We conducted several experiments on 14 ambiguous names using different parameters (context sizes, number of topics, corpora for training the topic model). For evaluating our system we created a silver standard corpus that does not need any manual annotation but exploits the internal link structure of Wikipedia.

We showed that our system outperforms the two baselines (majority class and Jaccard index) in many cases, especially for (i) entities for which not much training material is available, and (ii) entities with little surrounding context. We also showed that with our approach we can achieve high recall results, which is important for many applications, for example, search engines and QA systems.

5. Informative sentiment summaries

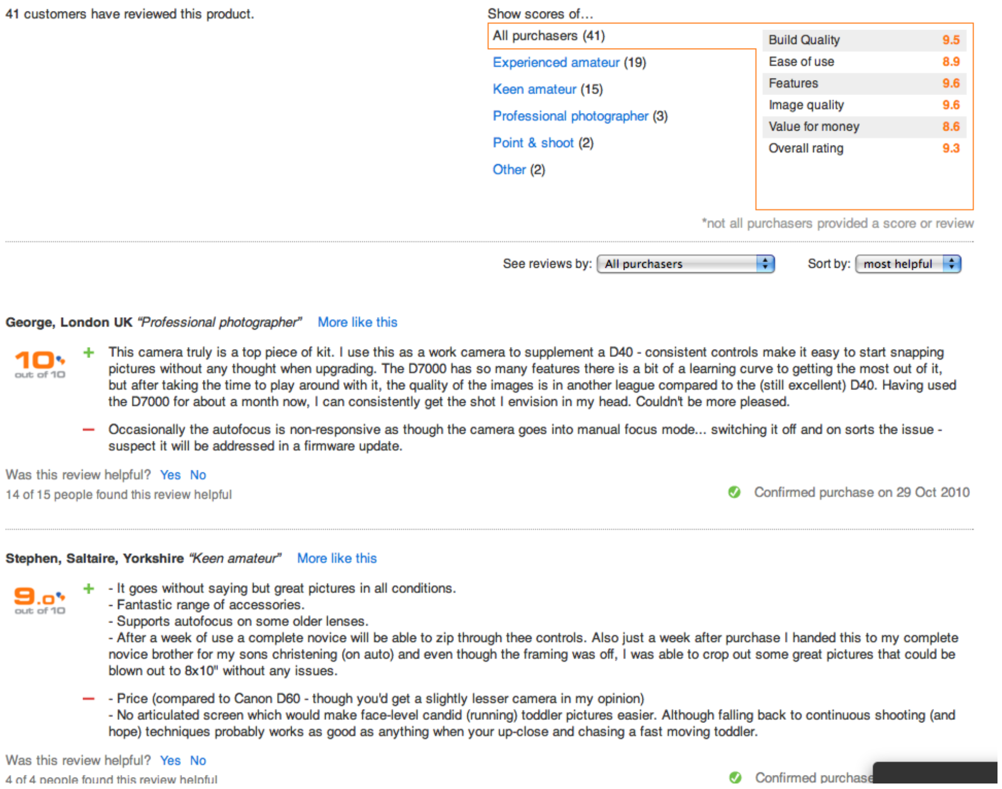
5.1. Motivation

In previous chapters (Chapter 3 and Chapter 4) we focused on techniques to improve retrieval of entities to obtain better retrieval results in general as well as for lesser known entities. However, good techniques to obtain the most relevant results are only one part of a good retrieval system. Another important part is to display the information in a way that allows users to quickly find the information they are looking for. This is especially important when there are hundreds or more possible results for a given search rather than only a few. A ranking algorithm can try to sort results by relevance to the user's search. However, this is not without problems. For example, it is not always obvious which of the results are most relevant. Another case is when the user is looking for very specific information that cannot be captured easily by a ranking algorithm.

In this chapter, we focus on the problem of displaying the most informative (i.e., the most relevant) information in an effective way[54]. We address this problem in the context of reviews because this is a fast-growing area of content that benefits from improved displaying techniques. With the rapidly growing amount of user-generated reviews, it is important for both companies and consumers to quickly assess what consumers think about a product they bought. For this purpose, some websites offer small summaries of reviews. These summaries can be very short and do not necessarily show the most important topic of the review. Other sites collect opinions about certain aspects of a product and present the results as a list.

Figure 5.1 shows reviews written on the platform `reevoo.com`. The pros and cons are not automatically extracted from review texts. Instead, when writing a review users are shown two text fields, one for positive feedback and one for negative feedback. They can enter text either in full sentences like in the first review, as a list of keywords, or as a mix of both like in the second review. Points can be given to a fixed set of features such as the build quality, ease of use, or image quality. Splitting reviews into positive and negative feedback allows users to see more easily what other users think is good or

[54]The approach presented in this chapter was published in (Glaser and Schütze, 2012).



41 customers have reviewed this product.

Show scores of...

All purchasers (41)
Experienced amateur (19)
Keen amateur (15)
Professional photographer (3)
Point & shoot (2)
Other (2)

Build Quality	9.5
Ease of use	8.9
Features	9.6
Image quality	9.6
Value for money	8.6
Overall rating	9.3

*not all purchasers provided a score or review

See reviews by: All purchasers   Sort by: most helpful

George, London UK *"Professional photographer"* More like this

10 out of 10

+ This camera truly is a top piece of kit. I use this as a work camera to supplement a D40 - consistent controls make it easy to start snapping pictures without any thought when upgrading. The D7000 has so many features there is a bit of a learning curve to getting the most out of it, but after taking the time to play around with it, the quality of the images is in another league compared to the (still excellent) D40. Having used the D7000 for about a month now, I can consistently get the shot I envision in my head. Couldn't be more pleased.

– Occasionally the autofocus is non-responsive as though the camera goes into manual focus mode... switching it off and on sorts the issue - suspect it will be addressed in a firmware update.

Was this review helpful? Yes No
14 of 15 people found this review helpful

Confirmed purchase on 29 Oct 2010

Stephen, Saltaire, Yorkshire *"Keen amateur"* More like this

9.0 out of 10

+ - It goes without saying but great pictures in all conditions.
- Fantastic range of accessories.
- Supports autofocus on some older lenses.
- After a week of use a complete novice will be able to zip through thee controls. Also just a week after purchase I handed this to my complete novice brother for my sons christening (on auto) and even though the framing was off, I was able to crop out some great pictures that could be blown out to 8x10" without any issues.

– - Price (compared to Canon D60 - though you'd get a slightly lesser camera in my opinion)
- No articulated screen which would make face-level candid (running) toddler pictures easier. Although falling back to continuous shooting (and hope) techniques probably works as good as anything when your up-close and chasing a fast moving toddler.

Was this review helpful? Yes No
4 of 4 people found this review helpful

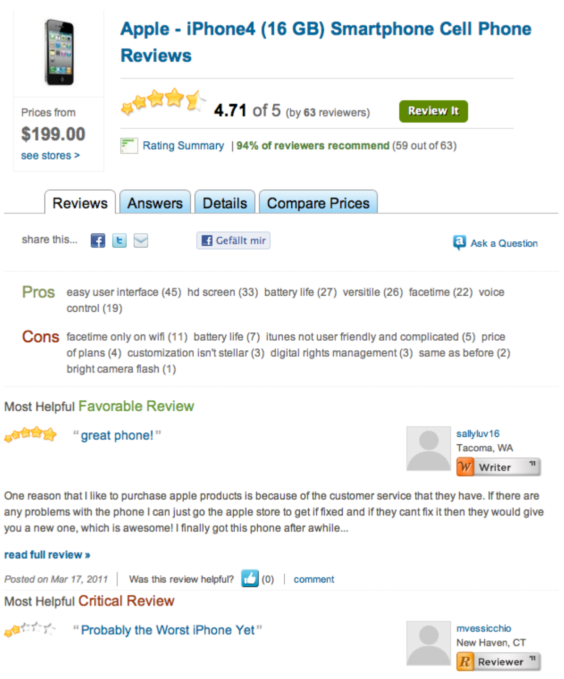
Confirmed purchase [redacted]

Figure 5.1.: Pros and cons in user reviews on `reevoo.com` (screenshot taken on June 28, 2011).

bad about the product. However, if reviews are long and not formatted as keywords, users still cannot quickly assess the information.

The website `viewpoints.com` also offers pros and cons, but in a different way, as can be seen in Figure 5.2. At the time of writing a review, users can select pros and cons that apply to their experience from a list provided by the website. Additionally, users can add their own pros and cons. The top of the product page contains a summary of pros and cons users chose and added themselves, taken from all reviews of the product and sorted by how often users picked these pros and cons. In the newer version of the website (last accessed on June 4, 2019) there is an additional option called "best uses", which works the same as the pros and cons list. Pros, cons, and best uses are also present in the individual reviews in the newer version.

Providing lists of pros and cons results in a good first overview of a product. However, one key problem that these approaches share is that they often only take the sentiment of the author into account and do not necessarily contain a reason why the product or aspect is rated positively or negatively. For example, the aspect *"battery life"* appears in both the pros list (with an occurrence count of 27) as well as in the cons list (with an occurrence count of 7) in Figure 5.2a (and with a similar distribution in Figure 5.2b).



Apple - iPhone4 (16 GB) Smartphone Cell Phone Reviews

Prices from $199.00 see stores >

4.71 of 5 (by 63 reviewers) Review It

Rating Summary | 94% of reviewers recommend (59 out of 63)

Reviews | Answers | Details | Compare Prices

share this... Gefällt mir Ask a Question

Pros easy user interface (45) hd screen (33) battery life (27) versitile (26) facetime (22) voice control (19)

Cons facetime only on wifi (11) battery life (7) itunes not user friendly and complicated (5) price of plans (4) customization isn't stellar (3) digital rights management (3) same as before (2) bright camera flash (1)

Most Helpful Favorable Review

"great phone!"

sallyluv16
Tacoma, WA
Writer

One reason that I like to purchase apple products is because of the customer service that they have. If there are any problems with the phone I can just go the apple store to get if fixed and if they cant fix it then they would give you a new one, which is awesome! I finally got this phone after awhile...

read full review »

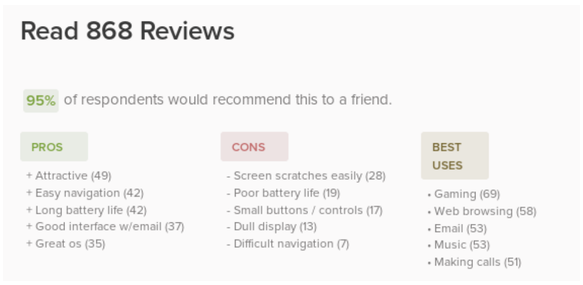
Posted on Mar 17, 2011 | Was this review helpful? (0) | comment

Most Helpful Critical Review

"Probably the Worst iPhone Yet"

mvessicchio
New Haven, CT
Reviewer

(a) summary part, individual review (2011)

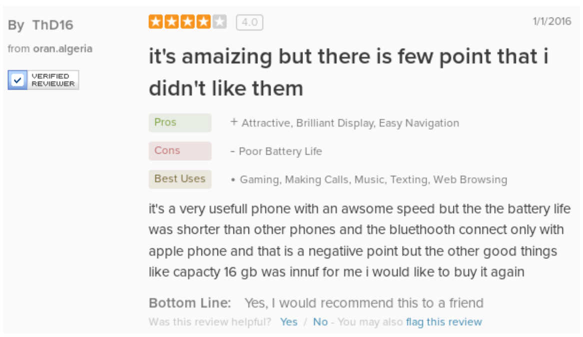


Read 868 Reviews

95% of respondents would recommend this to a friend.

PROS	CONS	BEST USES
+ Attractive (49)	- Screen scratches easily (28)	• Gaming (69)
+ Easy navigation (42)	- Poor battery life (19)	• Web browsing (58)
+ Long battery life (42)	- Small buttons / controls (17)	• Email (53)
+ Good interface w/email (37)	- Dull display (13)	• Music (53)
+ Great os (35)	- Difficult navigation (7)	• Making calls (51)

(b) summary part (2019)



By ThD16
from oran.algeria
VERIFIED REVIEWER

4.0 1/1/2016

it's amaizing but there is few point that i didn't like them

Pros + Attractive, Brilliant Display, Easy Navigation

Cons - Poor Battery Life

Best Uses • Gaming, Making Calls, Music, Texting, Web Browsing

it's a very usefull phone with an awsome speed but the the battery life was shorter than other phones and the bluethooth connect only with apple phone and that is a negatiive point but the other good things like capacty 16 gb was innuf for me i would like to buy it again

Bottom Line: Yes, I would recommend this to a friend

Was this review helpful? Yes / No - You may also flag this review

(c) individual review (2019)

Figure 5.2.: Summarized pros and cons from user reviews on `viewpoints.com` (screenshots taken on June 28, 2011 and June 4, 2019).

While more people rate the battery life as good, the people rating it as poor might have good reasons, which are not displayed in detail. Subsequently, a user who is trying to learn more details about the product has to read through several (possibly long) reviews to find this information.

In many applications of sentiment analysis, reasons are as important as the sentiment itself. For example, the marketing department of a smartphone manufacturer may want to go quickly through dozens of online forum posts by users and gather specific reasons as to why people dislike a smartphone in order to address these problems in the next product iteration. Statistics that count the number of positive and negative reviews (or the strength of sentment of these reviews) are less useful for this application.

This leads us to our third research question:

Research Question 3 How can opinions about entities be presented in an effective way so that users can see not only the sentiment (positive/negative) but also a reason the reviewer provides for the sentiment?

To address use cases of sentiment analysis like the one discussed, we present a system that aims to extract sentences that contain convincing reasons why the author thinks a product is good or bad. We call such a sentence a *supporting sentence* because it supports the author's assessment that the product is either good or bad. Supporting sentences contain convincing reasons that provide detailed positive or negative facts about the product or its aspects. Consider the following examples of what qualifies as supporting sentences:

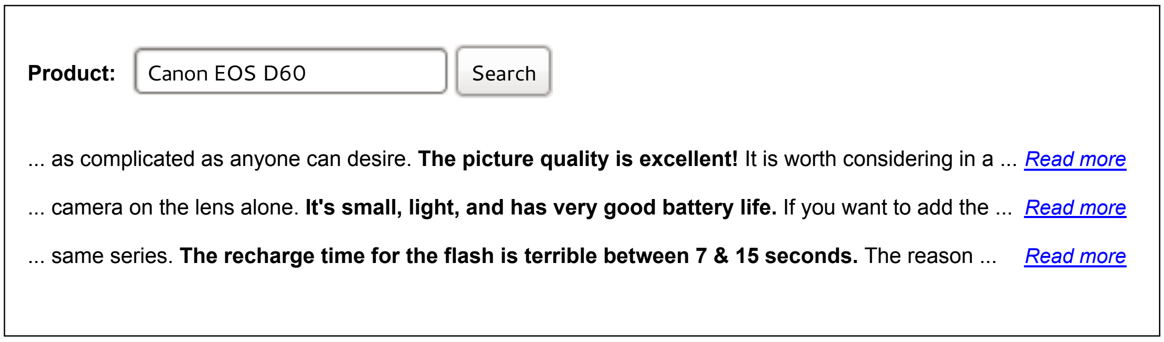
(5.1)
a. *"The picture quality is very clear and sharp."*
b. *"The battery life is only 2 hours."*
c. *"The zoom range is too limited."*
d. *"It has some great features like being able to reduce the file size photos stored on it."*
e. *"The only downside of this camera is the manual lens cover."*

Each of these sentences contains convincing reasons that provide detailed positive or negative facts about the product or its aspects. Non-supporting sentences, on the other hand, contain opinions without further details. The following examples list a few sentences that we consider non-supporting sentences:

(5.2)
a. *"I like this camera."*
b. *"This camera is not worth the money."*
c. *"So far, I like it very much."*
d. *"The delivery was fast."*

In example 5.2a, the opinion holder only says *"I like this camera"*, but without specifying what exactly they like about it. In example 5.2b a user is voicing a negative opinion about the camera but also without a specific reason. Very similar is example 5.2c, which has an additional problem of using a pronoun (*"it"*) with no reference to what the user likes. Example 5.2d seems to contain a reason, however, what was rated here has nothing to do with the product itself, but is an external factor (in this case, the website that ships the product). This means it is not a supporting fact for buying the product itself.

Convincing reasons can be found in subjective opinions as well as factual sentences. We focus on extracting one supporting sentence per document that characterizes the sentiment of the review because an ideal summary sentence captures both the sentiment and the supporting information of the review. With this supporting sentence, companies and consumers can quickly browse one self-contained element of the review's content and decide whether or not it is worth reading more of the review. Figure 5.3 shows an example of a simple interface that can display supporting sentences for a product search. The goal is to provide users with an easy way to assess information that contains sentiment as well as reasons for the sentiment so they can make informed choices quickly.



... as complicated as anyone can desire. **The picture quality is excellent!** It is worth considering in a ... Read more

... camera on the lens alone. **It's small, light, and has very good battery life.** If you want to add the ... Read more

... same series. **The recharge time for the flash is terrible between 7 & 15 seconds.** The reason ... Read more

Figure 5.3.: Simple representation of supporting sentences in a search engine scenario.

Evaluating convincing reasons is not an easy task and annotating a large dataset requires substantial effort. We define a novel evaluation methodology and explore in our work if a crowdsourcing service such as Amazon Mechanical Turk can be exploited to evaluate this type of problem.

In summary, we make the following contributions in this chapter:

- We present a system that extracts supporting sentences from reviews—a sentence that includes the sentiment of the overall review as well as a supporting fact for the writer's opinion—which can be used to quickly skim many reviews and find the information a user is looking for.
- We show how a crowdsourcing service like Amazon Mechanial Turk can be used to obtain a large amount of annotations without the need to hire experts and discuss design questions and quality control for using such a service.

The remainder of the chapter is structured as follows. In Section 5.2 we define our supporting sentences and present the approach that uses a 2-step model consisting of sentiment classification and a weighting function. Section 5.3 provides a detailed overview of our evaluation methodology, the crowdsourcing platform Amazon Mechanical Turk, our task design, as well as a discussion about quality control on crowdsourcing platforms. In Section 5.4 we first present the data that we use for our experiments and discuss the results. The section finishes with an error analysis. Related work is described in Section 5.5. We conclude the chapter with a summary in Section 5.6.

5.2. Methods

In this section we first define *supporting sentences* based on three premises. We then describe a 2-step model that uses these premises to obtain a supporting sentence from a document, consisting of two parts: (i) a sentiment classification to obtain sentences with strong sentiment and (ii) a weighting function that gives us the final supporting sentence based on earlier observations made when defining a supporting sentence.

5.2.1. Supporting sentences

The underlying idea of our approach is that when people give reasons for their opinion, they usually mention at least one word that refers to an aspect, component, or property of the product, or to an experience the user had while using the product. For example, reasons given for liking or disliking a digital camera can be associated with words such as *"zoom"*, *"picture quality"*, *"battery life"*, and *"macro focus"*. We do not include the product itself because it is too general. Consider the following sentences:

(5.3) a. *"The camera is good."*
b. *"The picture quality is good."*

A sentence like 5.3a does not contain a reason as to why the product (in this case a camera) is good. In contrast, the sentence 5.3b gives a specific reason for the sentiment expressed in the review and allows the reader to make an assessment as to whether he or she finds this reason convincing or not.

Therefore, we base our approach on the following three premises:

Premise 1 A good supporting sentence conveys both the document's sentiment and a supporting fact.

Although convincing reasons can appear in factual sentences, we decided to extract sentences that do contain sentiment. Let us consider the following factual sentence:

(5.4) *"I took approximately 80 to 90 pics using AA batteries with the display off."*

Sentence 5.4 can be seen as containing a reason. However, there is no positive or negative sentiment in the sentence. The person is simply stating that he or she took a certain amount of pictures. If a sentence only describes a fact about the product, or, for example, how the product was used as shown in sentence 5.4, it is not clear which overall sentiment is conveyed in the review. Thus, we want our supporting sentences to contain both a reason as well as the polarity of the review to be self-contained. Our simple baseline for selecting the supporting sentence for the review's sentiment is to choose the sentence with the strongest sentiment. Our evaluation will show that the task of selecting a sentence with strong sentiment is clearly different from the task of selecting a good supporting sentence.

Premise 2 Supporting facts are most often expressed by noun phrases.

Since convincing reasons for expressed sentiment have not been studied in detail before, there are no guidelines as to what constitutes a convincing reason or supporting fact. The following examples, which we considere supporting sentences, illustrate the complexity of the task:

(5.5) a. *"The* ***picture quality***$_N$ *is* ***very***$_{Adv}$ ***clear***$_{Adj}$ *and* ***sharp***$_{Adj}$*."*
b. *"The* ***battery life***$_N$ *is* ***only***$_{Adv}$ ***2 hours***$_N$*."*
c. *"The* ***zoom range***$_N$ *is* ***too***$_{Adv}$ ***limited***$_{Adj}$*."*
d. *"It has some* ***great***$_{Adj}$ ***features***$_N$ *like being able to* ***reduce***$_V$ *the* ***file size photos***$_N$ *stored on it."*
e. *"The* ***only***$_{Adj}$ ***downside***$_N$ *of this camera is the* ***manual***$_{Adj}$ ***lens cover***$_N$*."*

One thing all sentences have in common is the existence of at least one noun phrase, which represents the core of the reason in the sentence (i.e., the aspect the reason is about). We therefore assume that noun phrases are very important for conveying reasons. Noun phrases are not the only important words in a supporting sentence. Adjectives, adverbs, and verbs are often needed to correctly convey the reason for the sentiment expressed as can be seen in sentences 5.5a–5.5e.

While adjectives, adverbs, and verbs are often important when expressing the sentiment of a reason, it can be difficult to convey the core of the reason without the use of noun phrases that refer to a specific aspect, as illustrated in the following examples:

(5.6) a. *"So far, I* ***like***$_V$ *it* ***very***$_{Adv}$ ***much***$_{Adv}$*."*
b. *"It is* ***easy***$_{Adj}$ *to* ***use***$_V$*."*

Both sentences 5.6a–5.6b contain adjectives and verbs that express the sentiment of the sentence. However, it is not clear what the adjectives and verbs are describing as the noun phrases in the sentences are pronouns and reference is not given. Without resolving the pronoun *"it"*, it is not possible to know what the exact reason is. Therefore, we assume that reasons are most often expressed by noun phrases that directly refer to an aspect of the product. We call a noun phrase which does express a supporting fact a *keyphrase*.

Premise 3 Noun phrases that express supporting facts tend to be domain-specific; they can be automatically identified by selecting noun phrases that are frequent in the domain—either in relative terms (compared to a generic corpus) or in absolute terms.

As we have seen in previous examples, noun phrases that express a supporting fact are usually an aspect or component of the product. In contrast, the noun phrases in the following examples are not domain-specific:

(5.7) a. *"The **delivery**$_N$ was fast."*
b. *"I started having **problems**$_N$ yesterday."*

Sentence 5.7a contains a noun phrase about an external factor (*"delivery"*). The reason it expresses is not about the product itself, but about the shop that sent the product. Sentence 5.7b contains a very general noun (*"problems"*), but it does not specify what the exact problem is. Therefore, we assume that noun phrases that express supporting facts should be domain-specific.

With this assumption we might miss supporting sentences that are worded using non-domain-specific words only, but in an original way so that they do convey a reason. However, there is no simple way to detect whether a sentence containing such words does contain a convincing reason about the product or not. Moreover, most good reasons appear many times and can be captured when they are expressed with keyphrases.

5.2.2. 2-step model

Based on the three premises from the previous section, we adapt a 2-step model which selects a supporting sentence in two steps. The first step exploits sentence level sentiment classification to determine the n sentences with the strongest sentiment within every review. In the second step, a weighting function is used to select one of the n sentences as the best supporting sentence for the given review.

Step 1: Sentiment classification

As stated in premise 1, good supporting sentences should convey both the review's sentiment as well as a supporting fact. For this reason, we first classify all sentences to determine their sentiment. The polarity of sentences containing sentiment can be positive (e.g., *"The zoom is great"*), negative (e.g., *"The pictures are all blurry"*), or neutral (e.g., *"I bought the camera yesterday"*). Reviews can consist of positive or negative sentences only, or they can consist of a mix of different sentiments. We use a sentiment classifier to classify every sentence in each review as positive or negative. We do not classify sentences as neutral because neutral is usually seen as "no opinion" (Liu, 2015) and as such does not provide why someone liked or did not like something.

The classifier assigns a confidence score to each classified sentence, which states how confident the classifier is about the classification result for this sentence. We use this confidence score to obtain two lists of ranked sentences for each review, one ranked list for positive sentences and one ranked list for negative sentences. We assume that the higher a confidence score is the stronger the sentiment is. The sentence with the highest positive confidence score has the strongest positive sentiment and the sentence with the highest negative confidence score has the strongest negative sentiment.

We then select the n sentences with the highest confidence value where the polarity of the sentences corresponds to the polarity of the entire review. For example, if the review's polarity is positive, we select the n positively ranked sentences with the highest confidence score. We choose sentences that have the same polarity as the entire review because a good supporting sentence conveys the sentiment of the review as stated in premise 1. We make an exception in cases where fewer than n sentences were classified with the review's sentiment. In these cases we take all m sentences with the review's polarity and fill the remaining $n - m$ slots with sentences of the opposite polarity to obtain a consistent number of n sentences for each review. For the sentences with the opposite polarity we choose the sentences with the lowest confidence score because it is likely that the classifier missclassified them and they actually belong to the opposite sentiment.

Step 2: Weighting function

We employ a weighting function based on premises 2 and 3 presented in Section 5.2 and score a sentence based on the number and types of noun phrases, which we will describe in more detail in this section.

Sentences with convincing reasons almost always contain at least one keyphrase related to the product. This keyphrase is usually a noun (e.g., *"battery", "lens"*) or a compound noun (e.g., *"battery life", "camera size"*). Compound nouns consist of at least two words with no structural upper limit for the number of words (Plag, 2003). While there are some compound nouns consisting of three or more words which can be considered good keyphrases (e.g., *"lithium ion battery", "sd memory card", "shutter release button"*), the number of these words is very small compared to nouns and smaller compound nouns. We therefore limit the number of words in compound nouns to two words.

There are different theories on how compound nouns are composed. For example, Levi (1978) groups compounds into classes of nominal compounds (e.g., *"peanut butter"*), nominalizations (e.g., *"dream analysis"*), and non-predicate noun phrases (e.g., *"electric shock"*). We do not consider different classes but define a compound noun as consisting of two common nouns, i.e., `NN+NN` combinations, while leaving the decision of how to tag compound nouns to a POS tagger.

Compound nouns are generally more informative and specific than single nouns as can be seen in the following examples:

(5.8) a. *"battery" vs. "battery life"*

b. *"zoom" vs. "zoom button"*

c. *"mode" vs. "night mode"*

(5.9) a. *"life" vs. "battery life"*

b. *"supply" vs. "power supply"*

In examples 5.8a–5.8c all nouns and compound nouns are considered keyphrases. However, the single noun phrases are less specific than the compound nouns. Each of the single noun phrases can refer to different aspects if it is not clarified. *"Battery"* (5.8a) can refer to, e.g., the battery life or the type of battery required, *"zoom"* (5.8b) can refer to, e.g., the zoom range, zoom quality, or the zoom button, and *"mode"* (5.8c) can refer to, e.g., different types of modes. The compound nouns, on the other hand, already describe these aspects in more detail.

There are also cases in which the head noun of a compound noun does not provide a specific reason while the compound noun does as can be seen in examples 5.9a–5.9b. Nouns such as *"life"* (5.9a) and *"supply"* (5.9b) are too general and not considered good keyphrases for our domain, while when modified by another noun they become useful keyphrases.

Based on these observations, we compute scores in a way that gives a higher weight to compound noun phrases than to simple noun phrases. This means that we have to separate nouns and compound nouns in the scoring function. Since not all noun phrases are keyphrases as shown in the examples in 5.9, we further investigated the matter by analyzing the noun phrases in our dataset.

Many highly frequent nouns are keyphrases correlated with reasons, but there are also frequent nouns like *"money"* or *"people"* that authors often use in product reviews but do not indicate a good reason. Our basic assumption is that relevant keyphrases occur frequently in a corpus of the corresponding domain and more rarely in a general corpus such as Wikipedia. We consider the distribution of terms in a general corpus like Wikipedia to be domain-independent and use it to determine the relative frequencies of noun phrases in our domain-dependent review corpus. Let $f_{dom}(p)$ be the domain-specific absolute (raw) frequency of phrase p and $f_{wiki}(p)$ the absolute frequency of p in the English Wikipedia. Keyphrases that do not occur in Wikipedia are discarded. The relative frequency of p is then definied as in Equation 5.1.

$$f_{rel}(p) = \frac{f_{dom}(p)}{f_{wiki}(p)} \tag{5.1}$$

Not all nouns with a high relative frequency are good keyphrases. There are other nouns that occur rarely in Wikipedia and therefore have a high relative frequency f_{rel}. Good keyphrases for reasons must have a high raw frequency in the domain in addition to a high relative frequency f_{rel}. We define keyphrases in our approach as those nouns where both values are above a certain threshold, i.e., we consider a phrase (noun or compound noun) to be of *high frequency* if it is one of the $k\%$ phrases with the highest $f_{dom}(p)$ and at the same time is one of the $k\%$ phrases with the highest $f_{rel}(p)$. The exact thresholds for both frequencies depend on the dataset and domain.

Selecting noun phrases above a certain threshold with frequency heuristics provides us with a good number of keyphrases. However, there are still good keyphrases below the threshold which cannot be separated easily from the non-keyphrase nouns. Instead of discarding these potentially good keyphrases, we assume that they can possibly contribute to the scoring but are less important than the high frequent noun phrases. We include these *infrequent* nouns and compound nouns which do not meet the high frequency criterion (i.e., without a high absolute/relative frequency) in our scoring function but give them a lower weight.

Based on the observations and statements above, we design a scoring function that is a sum of four variables, each with an individual weight: (i) number of simple nouns with high frequency, (ii) number of compound nouns with high frequency, (iii) number of infrequent simple nouns, (iv) number of infrequent compound nouns. We define four sets:

- F_1: set of nouns with high frequency
- F_2: set of compound nouns with high frequency
- I_1: set of infrequent nouns
- I_2: set of infrequent compounds nouns

Describing the weighting formally, we use the evaluation function $[\![\phi]\!]$:

$$[\![\phi]\!] \equiv \begin{cases} 1 & \text{if } \phi \text{ is true} \\ 0 & \text{otherwise} \end{cases} \tag{5.2}$$

We then define the score s of a sentence with n tokens $t_1 \dots t_n$ (where the last token t_n is a punctuation mark) as in Equation 5.3. It should be noted that each noun in a compound noun contributes twice to the overall score—once as part of the compound noun and once as a single noun.

$$
\begin{aligned}
s = \sum_{i=1}^{n-1} \quad & w_{f_2} \cdot [\![(t_i, t_{i+1}) \in F_2]\!] \\
& + w_{i_2} \cdot [\![(t_i, t_{i+1}) \in I_2]\!] \\
& + w_{f_1} \cdot [\![t_i \in F_1]\!] \\
& + w_{i_1} \cdot [\![t_i \in I_1]\!]
\end{aligned} \tag{5.3}
$$

We determine the weights w_{f_2}, w_{i_2}, w_{f_1}, and w_{i_1} using logistic regression. Our underlying hypothesis is that the sentence with the strongest sentiment (i.e., the sentence that was classified with the highest confidence in the previous step) often does not provide a good reason. Our experiments confirm this hypothesis to be true. To create a training set for the regression we take the two top scoring sentences of each review. These two sentences are the two classes in the regression problem (top ranked sentence vs. the sentence at rank 2). Since taking all sentences adds too much noise, we eliminate sentence pairs where the top sentence is better than the second sentence on almost all of the set counts (i.e., count of members of F_1, F_2, I_1, and I_2). The weights w_{f_2}, w_{i_2}, w_{f_1}, and w_{i_1} which are estimated by the regression, are then used to score sentences according to Equation 5.3.

The first step of the model provides us with a set of n sentences with the strongest sentiment, corresponding to the polarity of the entire review. In the second step, we apply the scoring function in Equation 5.3 to each of these sentences to obtain their sentence scores. The sentence with the highest score within a set is selected as the supporting sentence for this review. This sentence is now considered the sentence that is most informative for providing both the sentiment of the review as well as a convincing reason for this sentiment.

5.3. Comparative evaluation using Amazon Mechanical Turk

In this section we describe how we evaluate supporting sentences. We start by reviewing existing evaluation methods and motivating a novel evaluation methodology. We then introduce the crowdsourcing service Amazon Mechanical Turk and describe the task design in detail. Finally, we discuss quality issues that arise when using crowdsourcing services and how to address them to improve the quality of the annotations.

5.3.1. Evaluation methodology

In this work, a supporting sentence is essentially a short summary of a review. To evaluate text summaries, different methods have been employed in previous work. In general, evaluation methods can be divided into two categories, intrinsic evaluation and extrinsic evaluation (Jones and Galliers, 1996). *Intrinsic evaluation* measures the system's objective, i.e., the quality of the summary based on different criteria like fluency, readability, and coherence, or creating hand-edited summaries and comparing them to the system's output. The comparison is done with some measure of similarity such as word or n-gram overlap (e.g., Minel and Piat (1997); Brandow et al. (1995); Kupiec et al. (1995); Lin and Hovy (2003)). *Extrinsic evaluation*, on the other hand, measures the role of a system given the related task, i.e., it measures the performance when used for another task such as question answering or comprehension tasks (e.g., Mani and Bloedorn (1997); Jing et al. (1998); Tombros and Sanderson (1998); Hirao et al. (2001); Doran et al. (2004)).

A widely used metric for automatic summarization evaluation is *ROUGE* (Recall-Oriented Understudy for Gisting Evaluation) (Lin, 2004). ROUGE includes different types of evaluation metrics based on statistics such as n-gram overlap, longest common subsequences, and skip-bigrams. Another evaluation method uses so called *basic elements* (BEs) (Hovy et al., 2005, 2006a), minimal semantic units which can be combined with other BEs to form larger units with varying flexibility. Instead of comparing string similarity, it is also possible to compare information content, which is done with shared atomic information units called *factoids* (van Halteren and Teufel, 2003; Teufel and van Halteren, 2004). These factoits are created from the summaries in the dataset and are refined during processing these summaries. The *pyramid method* (Nenkova and Passonneau, 2004; Nenkova et al., 2007) argues that there is no single best human-written summary and that several such summaries for one document need to be taken and processed for evaluation.

These standard methods require a large annotation effort by creating one or multiple hand-crafted summaries for each document. Using a crowdsourcing service to obtain such summaries is likely not a good idea because the task is quite complex. Writing good summaries requires a skilled writer on a native speaker level. Another problem is that writing summaries takes time—the person has to read through the entire document first and decide on the most important points, then write down the summary. While there might be people who would do such a task on a crowdsourcing service, it is hard to find them and it is likely that people with less skills and interest will produce summaries of low quality.

Due to these problems we design a novel evaluation methodology, which has several advantages. It does not require hand-crafted model summaries which means it is much less expensive and does not require skilled writers. Instead, it provides a framework that supports direct use of human judgment on the task that is targeted.

The general idea of our evaluation method is based on using relative judgments instead of absolute jugdements. For example, the traditional way to evaluate information retrieval systems is to use absolute judgments. One way to do this is to assign each retrieved document to a category on a graded scale which describes how relevant the document is (e.g., Rees and Schultz (1967); Tang et al. (1999)). Different numbers of categories have been used before, which give a more coarse or more fine scale between "non relevant" and "very relevant". Only the relevance of the document itself is taken when assigned to a category, making the judgment independent of other documents.

Relative judgment methods, on the other hand, do not view a document independently when judging its relevance but take other documents into account to give a judgment in relation to them (e.g., Joachims (2002); Freund et al. (2003); Burges et al. (2005); Zheng et al. (2007); Cao et al. (2007)). The output can then be a ranked list of documents ordered by their relevance. Carterette et al. (2008) investigate the difficulty of both judgment methods and provide evidence that relative judgments (i.e., "document A is more relevant than document B") are easier to make than absolute judgments.

We base our evaluation methodology on relative judgments and design the task in such a way that it asks annotators to decide, given two sentences A and B, which one has the more convincing reason. Ideally, we would want to have access to an exhaustive ranking of all sentences of a review, ranked according to their quality as supporting sentences. However, this type of annotation would be very expensive to obtain. Producing a ranking of all sentences in a review in relation to the selected supporting sentence would require $m(m-1)/2$ judgments (where m is the length of the review), which would produce a very large annotation task. Therefore, we limit the judgments to one relative judgment per review, consisting of the selected supporting sentence and a baseline sentence. This allows us to obtain a dataset that can be annotated via a crowdsourcing service in a short time and at little cost.

5.3.2. Amazon Mechanical Turk

Many tasks can be automated with intelligent computer programs nowadays. Still, there are tasks which cannot be done by programs because they would produce too many errors or require considerable world knowledge to be completed satisfactorily, and as such can only be solved using human intelligence. Because of this, internet crowdsourcing services like Amazon Mechanical Turk[55] (MTurk), where tasks are outsourced to a crowd, have become very popular in recent years.

Amazon Mechanical Turk is a crowdsourcing marketplace where individuals or organizations (the *requesters*) can post such work tasks that require humans (the *workers*, also called *turkers*) to solve them. To post a large task or project it has to be broken down into small individual tasks, the so-called *HITs* (Human Intelligent Tasks). For example, if the project is to annotate a document using certain criteria, then each HIT could be the annotation of one sentence. The requester pays a certain reward for each completed HIT to a worker (e.g., \$0.10), depending on the complexity and time required to finish the task, plus a small fee to Amazon for using the service. If the requester is not satisfied with the quality of a worker's submission, he can reject it. In this case, nothing has to be paid and the HIT is posted again so another worker can work on it. HITs can be posted in parallel so that several workers can work on them simultaneously.

Requesters can design templates for HITs which contain detailed instructions about the task. Workers are not required to work on all HITs in a project. Some may choose to work only on a few HITs while others work on a larger amount or everything. Because of this, there can be many different workers submitting results. If workers only submit a few HITs they have less experience with the task and might submit low-level results. On the other hand, submitting many HITs does not necessarily mean that the quality is better because workers might not work carefully or just submit spam. We discuss the quality of MTurk annotations and quality control mechanisms in Section 5.3.4.

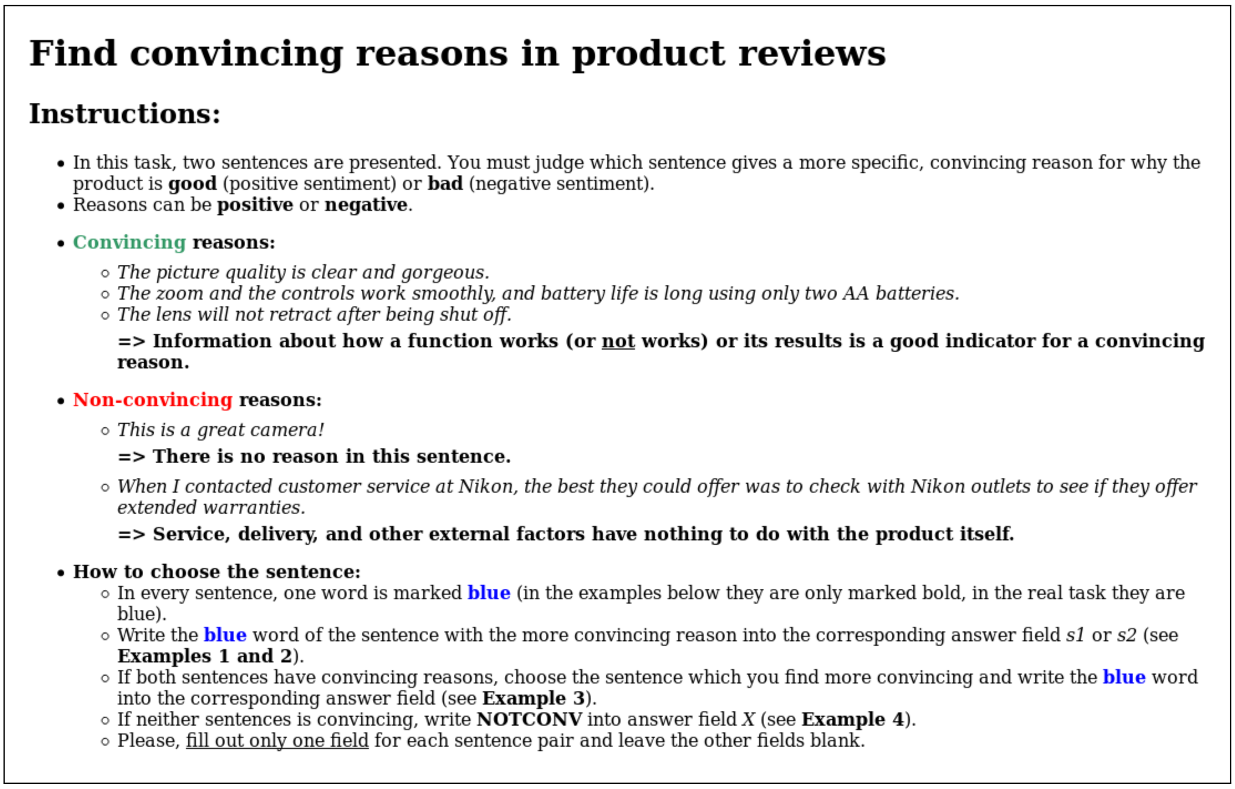
5.3.3. Task design

We created a HIT template including detailed instructions on how the workers should annotate the HIT, which can be seen in Figure 5.4. These instructions included definitions and examples of convincing reasons and non-convicing reasons, as well as examples of how to fill out the fields in order to complete the HIT (see Figure 5.5).

Each HIT consists of a pair of sentences. One sentence is the baseline sentence and the other one is the system sentence, i.e., the supporting sentence selected by the scoring function. We used a randomizer to shuffle the order of the baseline and system sentence in each HIT to avoid bias (e.g., workers always choosing the first or the second sentence). Instead of simply clicking the preferred sentence, workers had to fill out the answer fields correctly according to the instructions given in Figure 5.4. This was for quality control reasons, which we discuss in Section 5.3.4.

The workers were asked to compare both sentences and evaluate their relative quality.

[55]`https://www.mturk.com/`



Find convincing reasons in product reviews

Instructions:

- In this task, two sentences are presented. You must judge which sentence gives a more specific, convincing reason for why the product is **good** (positive sentiment) or **bad** (negative sentiment).
- Reasons can be **positive** or **negative**.
- **Convincing reasons:**
  - *The picture quality is clear and gorgeous.*
  - *The zoom and the controls work smoothly, and battery life is long using only two AA batteries.*
  - *The lens will not retract after being shut off.*

  => Information about how a function works (or not works) or its results is a good indicator for a convincing reason.
- **Non-convincing reasons:**
  - *This is a great camera!*

    => There is no reason in this sentence.
  - *When I contacted customer service at Nikon, the best they could offer was to check with Nikon outlets to see if they offer extended warranties.*

    => Service, delivery, and other external factors have nothing to do with the product itself.
- **How to choose the sentence:**
  - In every sentence, one word is marked **blue** (in the examples below they are only marked bold, in the real task they are blue).
  - Write the **blue** word of the sentence with the more convincing reason into the corresponding answer field *s1* or *s2* (see **Examples 1 and 2**).
  - If both sentences have convincing reasons, choose the sentence which you find more convincing and write the **blue** word into the corresponding answer field (see **Example 3**).
  - If neither sentences is convincing, write **NOTCONV** into answer field *X* (see **Example 4**).
  - Please, fill out only one field for each sentence pair and leave the other fields blank.

Figure 5.4.: Instructions for MTurk workers.

This was done by selecting one of the following three options:

1. Sentence 1 has the more convincing reason.
2. Sentence 2 has the more convincing reason.
3. Neither sentence has a convincing reason.

In some cases both sentences contain a convincing reason. An example of this is shown in example 3 in Figure 5.5. In these cases, the worker had to choose only one sentence which he considered to be more convincing, i.e., fill out only one field.

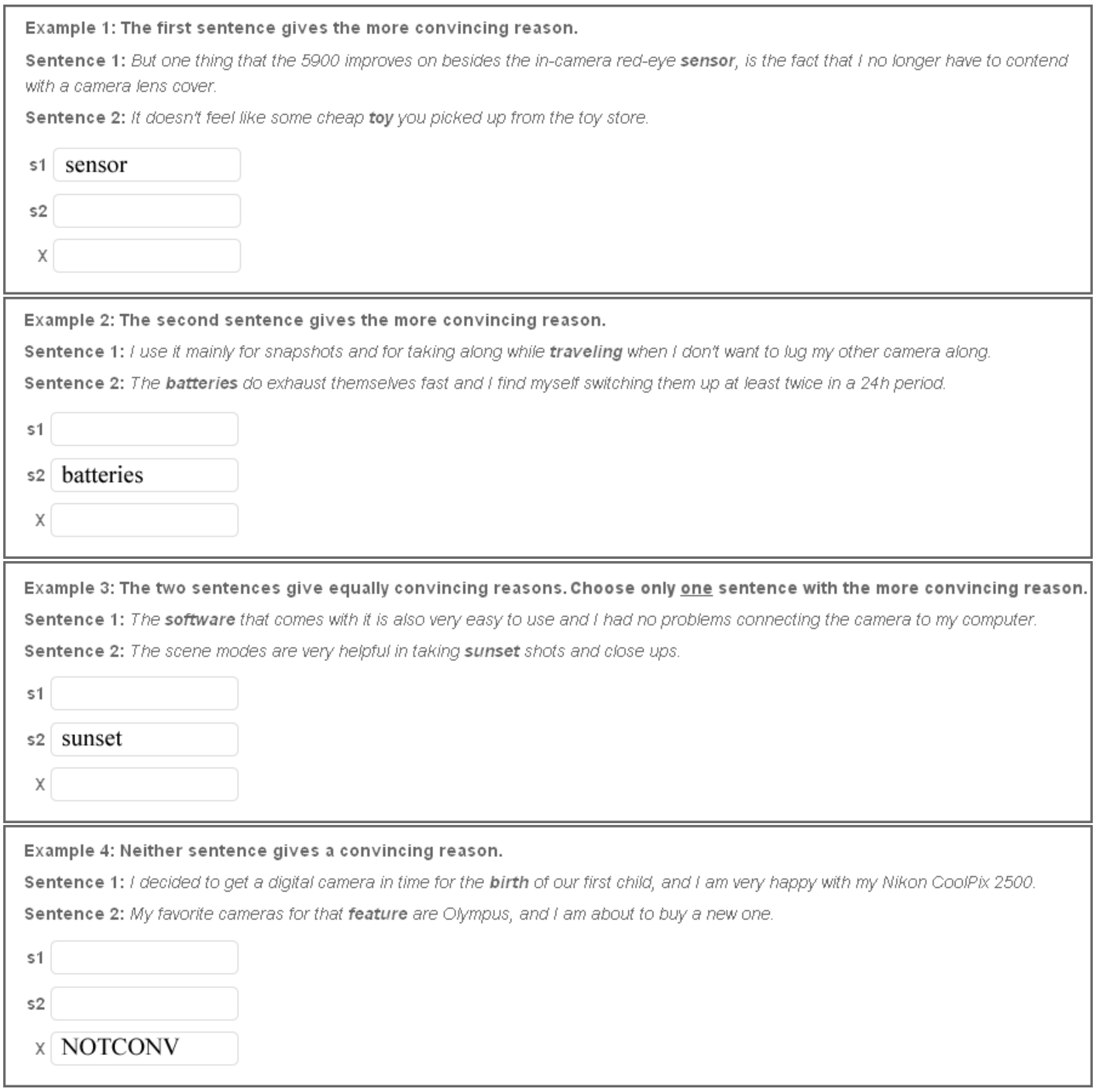
Each HIT can have multiple *assignments*. For example, if a HIT has three assignments then three different workers can submit answers. Every worker can process each HIT only once, i.e., it is not possible that one worker completes all assignments of one HIT. This mechanism is there to make sure that the assigments are always done by different workers. For each HIT we posted we created three assignments—to collect answers from three different workers which allows us to obtain annotator agreement.

Based on annotations submitted by the workers, we computed a score for each baseline sentence and system sentence as follows. For every answer of one HIT:

- **Add 1 to the baseline sentence score** if the worker thinks the baseline sentence contains the more convincing reason.

- **Add 1 to the system sentence score** if the worker thinks the supporting sentence from our system contains the more convicing reason.
- **Leave scores unchanged** if the worker thinks neither sentence contains a convincing reason.

This way, every baseline sentence and system sentence obtains a score between 0 (no worker found this sentence to be convincing) and 3 (all workers found this sentence to be convincing). The sentence with the higher score is selected as the best supporting sentence for the corresponding review.



Example 1: The first sentence gives the more convincing reason.

Sentence 1: *But one thing that the 5900 improves on besides the in-camera red-eye* ***sensor****, is the fact that I no longer have to contend with a camera lens cover.*

Sentence 2: *It doesn't feel like some cheap* ***toy*** *you picked up from the toy store.*

s1	sensor
s2	
X	

Example 2: The second sentence gives the more convincing reason.

Sentence 1: *I use it mainly for snapshots and for taking along while* ***traveling*** *when I don't want to lug my other camera along.*

Sentence 2: *The* ***batteries*** *do exhaust themselves fast and I find myself switching them up at least twice in a 24h period.*

s1	
s2	batteries
X	

Example 3: The two sentences give equally convincing reasons. Choose only one sentence with the more convincing reason.

Sentence 1: *The* ***software*** *that comes with it is also very easy to use and I had no problems connecting the camera to my computer.*

Sentence 2: *The scene modes are very helpful in taking* ***sunset*** *shots and close ups.*

s1	
s2	sunset
X	

Example 4: Neither sentence gives a convincing reason.

Sentence 1: *I decided to get a digital camera in time for the* ***birth*** *of our first child, and I am very happy with my Nikon CoolPix 2500.*

Sentence 2: *My favorite cameras for that* ***feature*** *are Olympus, and I am about to buy a new one.*

s1	
s2	
X	NOTCONV

Figure 5.5.: Example annotations with correct field fillings for MTurk workers.

In some cases, both sentences obtained the same score. For example, if one worker chose sentence 1, the second worker chose sentence 2, and the last worker chose neither sentence, then the final score for both sentences was 1. In such cases, we posted the corresponding HIT one more time, i.e., with one assignment. If this additional HIT did not break the tie, then we made no decision as to which sentence was better for this sentence pair. We labeled these sentence pairs "no decision" (N-D).

5.3.4. Quality control

Workers on Amazon Mechanical Turk are usually non-experts in the fields of the tasks they perform. They have different educational backgrounds and also different motivations for doing tasks. In the beginnings of MTurk, the majority of workers (70-80%) were from the United States, which was attributed to workers being required to have a bank account in the U.S. to receive money for their work (Ipeirotis, 2010a; Ross et al., 2010). When Amazon changed this policy and made it possible for Indian workers to receive cash, the population changed significantly (Ipeirotis, 2010a; Eriksson and Simpson, 2010; Paolacci et al., 2010). Studies about demographics found that money is not always the primary motivation and some workers use MTurk as entertainment in their free time (Paolacci et al., 2010; Horton et al., 2011; Ipeirotis, 2010a).

Spam is a well-known problem on Amazon Mechanical Turk. Laws et al. (2011) conducted preliminary experiments in which workers had to decide whether a document had a positive or negative sentiment by using simple radio buttons. Simply clicking a button attracted a great amount of spammers and lowered the overall accuracy to about 55%. Vuurens (2011) performed a test in which workers had to judge query-document pairs according to a 5-point rating scale. They identified proper workers (55%), sloppy workers (6%), and different types of spammers (together 39%). Ipeirotis et al. (2010) developed an algorithm that assigns a quality score to each worker. In their experiments they found that using this algorithm increased the overall annotation quality and that 30% of the original answers were submitted by spammers. Kaufmann et al. (2011) included content-related test questions in their experiments which had to be answered correctly. They filtered out 33.7% of the workers that were identified as spammers who did not answer these questions correctly.

The MTurk service offers some built-in anti-spam features and allows requesters to set certain constraints to combat spammers. One option that requesters have is to require workers to have a minimum acceptance rate, which they acquire through previously done HITs. A worker who constantly submits poor work will have his work rejected often, resulting in a low acceptance rate. Only allowing workers with an acceptance rate of, for example, 95% or higher and at least 1000 completed HITs to work on HITs increases the chance of getting good workers for the task. However, as mentioned by Eickhoff and Vries (2013), a high acceptance rate does not necessarily mean that the worker performs reliable and good work. One reason is that many submitted HITs are accepted by the requester and only discarded after some filtering process. These discarded HITs cannot be attributed to the worker retrospectively, resulting in giving him a higher acceptance rate than he should have. Another problem is that it is very simple and cheap to create a new account and boost the acceptance rate by working on your own HITs as described by Ipeirotis (2010b).

In 2011, Amazon introduced "Masters", a special qualification given to workers who demonstrate excellent performance over time and a wide range of HITs. The exact requirements of becoming a Master are not public. If a Master's performance decreases the Master status is retracted. Staffelbach et al. (2014) compared the quality of Master turkers to non-Master turkers and found that Master turkers submit less spam and have a higher return rate. However, since the number of workers who are Masters is not that high it takes much longer for a task to be completed if only Masters are allowed to work on it. Difallah et al. (2012) note that the Masters qualification is given by Amazon based on previously done work and requesters cannot decide who is a quality-worker. This means that many people might actually be very well qualified, at least for certain tasks, but are not allowed to work on "Masters only" HITs because they do not have the official qualification.

A requester can also filter out workers by location (country). This can help achieve better results, especially in language-specific tasks where it is important that native speakers work on the HITs. However, it does not hinder non-native speakers living in the accepted countries to do the HITs. Before working on the actual HITs, workers can be required to participate in a qualification test. This way they acquire training prior to the actual task and a requester can filter out unreliable workers before they work on possibly hundreds of HITs. However, an initial test might shy away potentially good workers from working on the task as it is additional work.

All of these methods can help improve the quality of the submitted work. However, these restrictions alone are generally not enough as they can all still be exploited. Difallah et al. (2012) give an overview of the limitations of these methods and describe possible attacks on tasks. Another problem is that the more restrictions a requester puts into place, the more the pool of possible workers is reduced. This means it takes much longer until a task is completed.

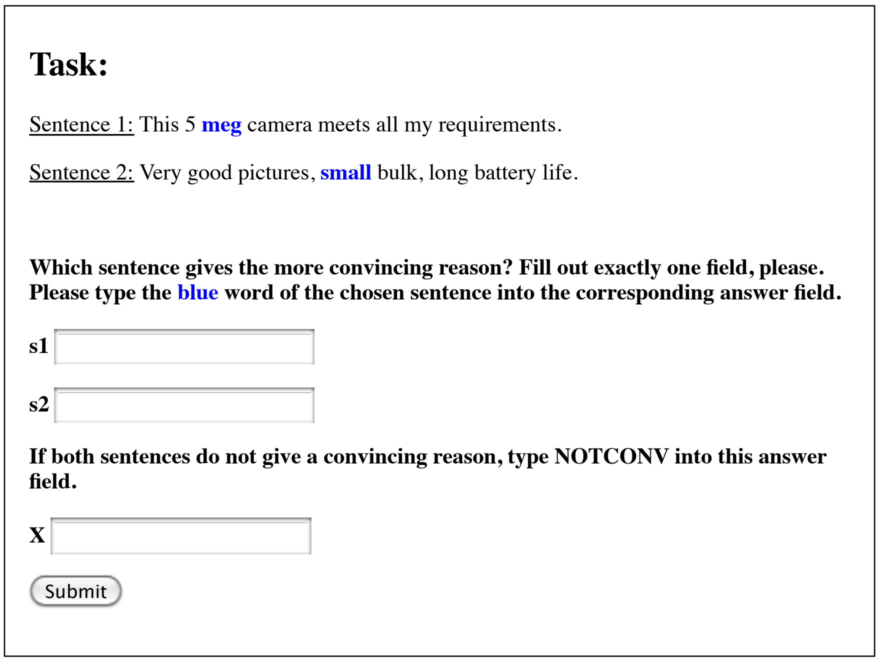
Besides using these quality control measures from Amazon, several other methods for preventing spam submissions have been used and discussed in previous work. A very common strategy is to collect and compare multiple responses per HIT submitted by different workers. Workers who constantly submit answers that differ from the majority vote of all answers are likely to be spammers. Snow et al. (2008) conducted tests to find out how many workers are necessary to obtain the same level of quality as with expert annotators. In an affective text analysis experiment where workers were asked to classify short headlines into one of six emotions, they found that depending on the complexity (some emotions were harder to classify than others) two to nine workers were needed to achive the same quality.

As Laws et al. (2011) noted, having a simple interface such as using only radio buttons attracts many spammers. Kittur et al. (2008) propose to include questions that have verifiable answers as part of the task, for example, how many images and sections an article presented in the HIT has. Spammers can then easily be identified as workers who constantly submit wrong answers. Paolacci et al. (2010) suggest including an ordinary question with the only possible answer "never" and filtering out all workers that fail to answer this question correctly because it is clear they did not read the question. A similar approach is to include some HITs with gold data (Sorokin and Forsyth, 2008). Since the correct answer is known for these HITs, workers who submit the wrong answer for them can be excluded and their work rejected.

Requesters do not only receive the answers of each worker but also some additional information for each completed assignment such as worker ID and the time each worker spent on the assignment before submitting it. This work time can be analyzed to identify spammers who spend considerably less time on an assignment as is needed to complete it (Kittur et al., 2008; McCreadie et al., 2010) or show suspicious pattern such as finishing each assignment in the same amount of time, i.e., they use an automated script or do not care about the task and just want to finish HITs (Zhu and Carterette, 2010). While this can filter out some spammers, it might miss spammers who are aware of this fact and know how to avoid being caught. As Downs et al. (2010) noted, spammers could hide their bad intentions by clicking through the task slower, while possibly doing other things at the same time and not paying full attention to the HITs.

Paying more money to workers for each HIT might lower the time until the task is completed, however it does not necessarily increase the quality of the submitted answers (Mason and Watts, 2010; Alonso and Baeza-Yates, 2011). As Le et al. (2010) noted, a high reward per HIT tends to attract unreliable workers.

To obtain high quality annotations for our dataset we used a combination of restrictions provided by the MTurk service and measures directly implemented into the task. We only allowed workers with an acceptance rate of 95% or higher and with at least 1000 completed HITs to work on our HITs. We did not use the Masters qualification because it was still very new at the time of our experiments and would have considerably reduced the size of the pool of possible workers for the task.



Task:

Sentence 1: This 5 **meg** camera meets all my requirements.

Sentence 2: Very good pictures, **small** bulk, long battery life.

Which sentence gives the more convincing reason? Fill out exactly one field, please.
Please type the blue word of the chosen sentence into the corresponding answer field.

s1

s2

If both sentences do not give a convincing reason, type NOTCONV into this answer field.

X

Figure 5.6.: Task interface for MTurk workers.

Figure 5.6 shows the task interface for workers which is located below the instructions (Figure 5.4) and examples (Figure 5.5) in every HIT. In each of the two sentences presented to the worker, two words were randomly marked in blue and bold. Workers were asked to type the blue word of the preferred sentence into the corresponding answer field $s1$ or $s2$. If workers found neither sentence to be convincing, they had to type `NOTCONV` into the special field X. This forced workers to actually read the sentences and spend some time typing the word, which would shy away spammers using simple browser scripts to autocomplete the HITs or spammers simply clicking radio buttons.

For each answer field we have a gold standard, namely the words we marked blue and the word `NOTCONV` for the special field. We used these gold standard answers to look for potential spam submitted by workers. We filtered all submitted answers that did not match with the gold standard and manually checked them. Our analysis showed that some workers mistyped some words. This indicates that the worker actually typed the word and not blindly copied it from the task. In Figure 5.7 we show a few examples of mistyped words and how we corrected them.

Some workers submitted inconsistent answers, for instance, they typed a random word or filled out all three answer fields instead of only one as requested in the tasks instructions. In such cases where it was obvious that the submitted answer was spam, we discarded the submission and reposted the HIT to receive a correct answer.

```
notcon   --> notconv          angel    --> angle
not conv --> notconv          com0pact --> compact
pdr-ml   --> pdr-m1           (per     --> per
```

Figure 5.7.: Example corrections of typographical errors in MTurk submissions.

After the task was completed, we counted how often every worker said that neither sentence is convincing (NOTCONV). Deciding whether a sentence contains a convincing reason is a subjective task. Even if there is a reason in the sentences, it might not be convincing for the worker. However, a high number of NOTCONV submission can indicate that the worker only copied the word for several sentence pairs without actually checking the content of the sentences. We also analyzed the time workers needed for every HIT. Since no task was done in less than 10 seconds, the possibility of just coyping the word without reading the sentences was rather low.

Since our evaluation methodology of asking workers of a crowdsourcing service to do relative judgments is novel, analyzing the annotation they performed is important in order to see if it is done consistently and reproducibly. Previous work shows that using a crowdsourcing service is a viable alternative for relevance assessments. Alonso and Mizzaro (2009) ask workers on MTurk to annotate documents as relevant or non-relevant and compared the agreement between workers and trained TREC assessors. They find that the quality when using MTurk is as good as when using expert annotators. Marge et al. (2010) use MTurk to transcribe and annotate samples of a speech corpus of meetings, and compute several kappa agreement scores (expert vs. expert, expert vs. turker, turker vs. turker). The scores were computed between every possible pair of annotators. They found that the agreement scores between experts is the highest (0.4) and the agreement score between experts and turker the lowest (0.19). The score between turker was in the middle (0.28).

The Fleiss' κ agreement score (Fleiss, 1971) for our final experiment is 0.17[56]. There are several reasons for this low κ score. MTurk workers were not trained to do the annotation task. They were only shown the instructions in the MTurk interface. It is possible that some workers did not fully understand the task but worked on it nevertheless (or they did not know that they did not understand it fully). Workers misunderstanding the task can decrease the quality of the answers and hurt the agreement score. Additionally, it is possible that some MTurk workers were not interested in producing quality answers and submitted random answers, which we could not catch with our spam analysis.

The biggest factor is probably the highly subjective nature of the task. Ranking

[56]What we compute is the agreement among the three workers that rated a HIT. The actual number of workers in our experiments was 61.

sentences according to how convincing the reasons within them are can be quite subjective. A sentence might contain a reason, but depending on the worker it might not be convincing. If both sentences contain reasons, then two workers might have different preferences of which reason is more convincing. This is in line with results presented by Nowak and Rüger (2010), who analyze the reliability of annotations obtained via crowdsourcing in the setting of multi-label image annotation. While they obtain high agreement scores among MTurk workers for objective concepts (e.g., *"clouds", "flowers", "snow"*), the agreement scores for subjective concepts (e.g., *"aesthetic", "quality", "fancy"*) are very low. Their study also reveals that even experts disagree more often on subjective concepts than on objective concepts, which results in a lower agreement score. This shows that subjective tasks are generally hard to solve which results in a lower κ score.

5.4. Experiments

In this section we present our experiments and results. We start by giving an overview of the data we use in Section 5.4.1. We then describe the experimental setup in Section 5.4.2 including the preprocessing of the dataset, the sentiment classification, how we determined frequencies and weights for the scoring function as well as our baseline. In Section 5.4.3 we present and discuss our results. We conclude with giving an error analysis in Section 5.4.4.

5.4.1. Data and preprocessing

Since the task we address is new, namely the exploration of supporting sentences in reviews, no existing labeled evaluation sets were available and we created our own dataset. We used part of the Amazon Product Review Data[57] from Jindal and Liu (2008). The whole dataset consists of more than 5.8 million reviews of several products, written by consumers and published on the `amazon.com` website. For our experiments we used the digital camera domain. We chose 17 different digital camera brands and extracted 1,136 potential products for these brands from the product information file included in the Amazon corpus data. The list contains some non-camera products (such as adapters and cases), which we filtered out manually. For the remaining 740 products we then extracted the corresponding reviews from the actual review data, resulting in a total of 15,340 reviews. Some products were only listed in the product information file but did not have a corresponding review. We discarded all products without corresponding

[57]`http://liu.cs.uic.edu/download/data/`

reviews. Statistics of the dataset before and after the filtering process as well as after further preprocessing steps are presented in the next section in Table 5.1 and Table 5.2.

Authors can give an overall rating to the product when writing a review for Amazon. This overall rating is represented by a number of stars on a scale of five. The possible ratings an author can give are 5 (*"very positive"*), 4 (*"positive"*), 3 (*"neutral"*), 2 (*"negative"*), and 1 (*"very negative"*). It is not possible to give half stars. These star ratings can be used as polarity labels for the reviews. We did not make fine-grained polarity distinctions (e.g., *"positive"* and *"very positive"*) in our experiments. Therefore, we unified star ratings of 4 and 5 to *"positive"* and star ratings of 1 and 2 to *"negative"*. We discarded all reviews with a star rating of 3. These reviews often contain both positive and negative sentences equally, which makes classification hard.

Preprocessing

The Amazon dataset provides reviews as raw text with no linguistic annotations. In a first preprocessing step we used the text annotation tool TreeTagger (Schmid, 1994, 1995) with the English parameter file, trained on the Penn Treebank (Santorini, 1990), to tokenize the text and tag it with part-of-speech (POS) tags. Sentence-ending punctuation marks (e.g., `.!?`) are tagged with the POS tag `SENT` by the TreeTagger. We used this POS tag to split each review into individual sentences.

The reviews from the Amazon dataset are user-generated content, i.e., not written by professional writers but by ordinary people. Such user-generated content has the problem that it is often written in incoherent English without proper grammar, spelling, and punctuation. Some users also like to add smilies to their content or use special characters for styling purposes, for example, underscores to mark words or phrases as underlined (e.g., *"This is the _best_ camera"*) and asterisks to mark words or phrases as bold (e.g., *"This is the *best* camera"*). Sometimes, people use a tilde as the end-of-sentence punctuation mark (e.g., *"let's see what happens∼"*). The tilde can be mostly seen in casual messages posted on social media such as facebook or twitter (Rudrapal et al., 2015), however, we found some occurrences of it in our data as well.

These extra characters can lead to incorrectly split tokens, which in turn yield wrong POS tags. After running TreeTagger the first time on the reviews we analyzed the output to see which problems arised. We then corrected the most frequent problems by cleaning the dataset before running TreeTagger again on the cleaned dataset.

In particular, we performed the following steps to clean the dataset:

- Separate word-punctuation clusters[58] such as `word...word` and `word/word`

[58]Since there is no whitespace the tokenizer considered them as one token.

- Remove stylistic marks such as `_`, ∗, and ∼ as in `_word_`, ∗`word`∗, and `word`∼
- Remove smileys such as `:-) ;-) :) ;) =) :D :(` etc.
- Remove HTML tags such as `<br> <br />`

TreeTagger sets sentence boundaries after every end-of-sentence mark (tagged with POS tag `SENT`), which results in incorrectly detected sentences if users write sentence-ending punctuation inside parantheses or use more than one sentence-ending punctuation, as can be seen in the following examples:

(5.10) (a) `<s>I like the camera ( and its functions !</s> <s>) .</s>`

(b) `<s>I like the camera !</s> <s>! </s> <s>!</s>`

To remove such incorrectly detected sentences we delete all sentences with three or less tokens. This also removes "sentences" consisting of only names (like the signature at the end of a review) and others which cannot be considered as sentences.

Some reviews do not end with an end-of-sentence marker, either because the user did not put a punctuation mark or used a tilde as replacement, which was removed in a previous step. In such cases, we added a period at the end to obtain the correct POS tag for end of sentence, which we need for separating the review into individual sentences.

We excluded all short reviews that have four or less sentences. We found that these short reviews are often of low quality because they were written hastily and without much care or detail. They are often full of grammatical and typographical errors, and do not give good reasons.

Table 5.1 shows statistics of the 17 camera brands we used for our experiments. The second column, Prod(A), shows the number of all products we extracted for each brand from the product information file. The next column, Prod(-E), shows the number of products after excluding all the products which do not have a corresponding review. The remaining three columns present the number of documents (reviews) in different stages of the data creation and cleaning process. Docs(-E) is the number of total reviews we extracted for each product that has corresponding reviews, a total number of 15,340 documents covering a total of 740 products. Docs(-3s) shows the number of documents we obtained after we discarded reviews with a star rating of 3. The total number of reviews after this step is 14,174, which is 92.39% of the original amount. Docs(>4) is the final number of documents after cleaning the corpus and removing short reviews with four or less sentences. The final total number is 11,624 documents, which is 82.02% of the reviews before cleaning and 75.78% of the original number of extracted reviews.

We split the final dataset of 11,624 documents into a training set (85%) and a test set (15%). The number of documents and the number of sentences in each part can be

Brand	Prod(A)	Prod(-E)	Docs(-E)	Docs(-3s)	Docs(>4)
AGFA	9	7	96	82	61
Canon	108	59	2,462	2,300	1,870
Casio	66	43	568	508	416
EasyShare	20	7	192	178	154
Fujifilm	91	54	1,354	1,260	1,029
HP	98	61	827	741	597
Kodak	139	91	2,406	2,252	1,829
Leica	16	5	31	29	27
Mustek	25	12	35	33	23
Nikon	83	47	1,463	1,327	1,099
Olympus	115	86	2,271	2,091	1,536
Panasonic	64	47	501	479	415
Pentax	48	37	368	338	287
Polaroid	36	14	112	96	62
Samsung	88	52	196	180	131
Sony	106	99	2,307	2,150	1,473
Toshiba	24	19	151	128	108
All	1,136	740	15,340	14,172	11,624

Table 5.1.: Statistics of 17 different camera brands used for the dataset.

seen in Table 5.2. The table also lists other statistics of our dataset. A review has an average number of 13.36 sentences, the median number of sentences is 10.

5.4.2. Experimental setup

Sentiment classification

We used the Stanford maximum entropy classifier[59] (Manning and Klein, 2003), version 2.1.1, and trained it on the sentences in the training set. The Stanford maximum entropy classifier is a probabilistic classifier which provides us with a probability distribution over the class assignment, thereby revealing how confident the classifier is in its decisions. We used a simple bag-of-words representation of the documents as features and removed punctuation and frequent stop words.

For the input of the binary classifier we used the unified labels *"positive"* (star rating of 4 or 5) and *"negative"* (star rating of 1 or 2). Since authors give the star ratings for the entire review, these labels are only available at the document level. To classify at the sentence level, we need sentence level sentiment labels. Manually annotating such a large amount of data would have been too expensive of a task. Therefore, we assumed

[59]`https://nlp.stanford.edu/software/classifier.shtml`

Type	**Number**
Brands	17
Products	740
Documents (all)	15,340
Documents (cleaned)	11,624
Documents (train)	9,880
Documents (test)	1,744
Average number of sents	13.36
Median number of sents	10

Table 5.2.: Key statistics of our dataset.

that if a review has a positive (resp. negative) label then most sentences convey a positive (resp. negative) sentiment and propagated the labels from the document level to the sentence level. This means all sentences occurring in positive reviews were labeled as *"positive"* and all sentences occurring in negative reviews were labeled as *"negative"*. The accuracy of the classifier is 88.37% on propagated sentence labels.

We then applied the trained classifier to the sentences in the test set. For each review we selected the n sentences classified with the same sentiment as the document level polarity and with the highest probability score given by the clasifier. We chose $n = 5$, i.e., if the document was positive (resp. negative), we selected the 5 sentences with the highest positive (resp. negative) probability score. If there were only m (with $m < 5$) sentences with the corresponding document polarity, we added the $(n - m)$ sentences classified with the opposite polarity and the lowest probability score to always obtain a set of n sentences. A low confidence value for these sentences might indicate that the classifier made the wrong decision and the sentences actually have the correct polarity. Due to the labeling scheme of the training input such errors are more likely to occur. We used this subset of n sentences for further processing using the scoring function in Section 5.2.2, Equation 5.3.

Determining frequencies and weights

In this section we discuss how we determined the frequencies and weights for the scoring function that we described in Section 5.2.2. To obtain the absolute frequency of nouns and compound nouns we computed their token frequency in the training set. The relative frequency was then computed as described in Section 5.2.2, Equation 5.1, using the English Wikipedia[60] (April 5, 2011 version) as the general corpus.

We analyzed the extracted noun phrases and compound noun phrases for both absolute and relative frequencies and found a higher percentage of highly frequent qualitative keyphrases for compound nouns compared to single noun phrases in our corpus. This is not surprising as compound nouns are more specific while a highly frequent single noun phrase can refer to something non-domain specific (e.g., *"time"*, *"thing"*, *"year"*, *"money"*). A list of the top 50 frequent nouns and compound nouns for both absolute and relative frequencies can be found in Appendix C in Table C.1 and Table C.2.

When looking at the top nouns of relative frequency (cf. Appendix C, Table C.2) we noticed a few words that contained typographical errors (e.g., *"camrea"*, *"battries"*, *"quailty"*, *"stablization"*). We only computed the relative frequency of words which appear in Wikipedia, otherwise we discarded them. The reason why there were still some words with typographical errors in our list is that these wrongly spelled words also exist in Wikipedia. We found that many of these misspelled words are good keyphrases related to our domain, so we did not manually filter them out.

With these observations we define our sets of nouns and compound nouns as follows:

- F_1: set of nouns in the top $k_n = 2.5\%$ for both absolute and relative frequencies
- F_2: set of compounds in the top $k_c = 5\%$ for both absolute and relative frequencies
- I_1: set of nouns not in the top $k_n = 2.5\%$ for either absolute or relative frequencies
- I_2: set of compounds not in the top $k_c = 5\%$ for either absolute or relative frequencies

Using these thresholds we obtain a high amount of good keyphrases with only few false positives. There are still other good keyphrases below the thresholds, however, it is not easy to separate them from the non-keyphrases.

Phrase	**Param.**	**Value**
keyphrase compounds	w_{f_2}	1.07
non-keyphrase compounds	w_{i_2}	0.89
keyphrase nouns	w_{f_1}	0.46
non-keyphrase nouns	w_{i_1}	0.01

Table 5.3.: Weight settings.

We used logistic regression to determine the values for the parameters w_{f_2}, w_{i_2}, w_{f_1}, and w_{i_1} of the scoring function (see Table 5.3). The values indicate the relative importance of the four different types of terms. Compound nouns are more important than

[60]wiki dump enwiki-20110405

simple nouns as they provide more detailed information. Even non-keyphrase compound nouns below the threshold (w_{i_2}) provide more information, which is why their value is higher than the value of keyphrase nouns (w_{f_1}). Non-keyphrase nouns are not very important which is reflected in the very small weight assigned to them ($w_{i_1} = 0.01$).

Baseline

We define our baseline (BL) as the sentence with the strongest sentiment which is in alignment with the overall sentiment of the document. The idea behind this baseline is that if a sentence contains strong sentiment, it should be an important sentence in the document. We use the output of the sentence classifier described in Section 5.2.2 to obtain the baseline: the sentence with the highest confidence score (of the same polarity as the overall polarity of the document) is used as the baseline sentence.

5.4.3. Results and discussion

We applied the scoring function in Section 5.2.2, Equation 5.3 using the parameter values listed in Section 5.4.2, Table 5.3 to the $n = 5$ highest scoring sentences we obtained with the sentiment classifier. Out of the 1,744 sentence pairs there were 364 cases (20.87%) in which the selected supporting sentence was the same as the baseline sentence. We removed these sentences pairs from our test set.

For each of the remaining 1,380 sentence pairs we created a HIT on MTurk with three assignments so that each HIT could be evaluated by three different workers. The baseline sentence and system sentence were scored according to our definition in Section 5.3. In 203 cases the result ended in a tie because the workers could not reach an agreement as to which of the sentences was more convincing. In a second pass, we reposted these 203 sentence pairs with one assignment per HIT, to break the tie.

Table 5.4 shows the agreement patterns after the first pass (all 1,380 sentence pairs), the second pass (the 203 sentence pairs without a decision in the first pass), and the final result (replacing the results of the 203 no-decision sentence pairs from the first pass with the results of the second pass). The first column shows the agreement patterns of HITs, depending on whether the three workers assigned to the HIT chose the baseline sentence (1), the system sentence (2), or neither sentence (4). For example, an agreement pattern of (1, 1, 2) means that two workers (not necessarily the first two workers) found the baseline sentence to be more convincing, and that one worker found the system sentence to be more convincing.

The top part of Table 5.4 shows the four cases of agreement which resulted in the baseline sentence to be choosen as the preferred sentence (BL>SY). The middle part

Agreement pattern	First pass (1,380 docs)	Second pass (203 docs)	Final result (1,380 docs)
(1, 1, 1)	75	0	75
(1, 1, 2)	149	68	217
(1, 1, 4)	80	0	80
(1, 4, 4)	74	27	101
BL>SY	378	95	473
(2, 2, 2)	247	0	247
(2, 2, 1)	240	67	307
(2, 2, 4)	182	0	182
(2, 4, 4)	130	26	156
SY>BL	799	93	892
(1, 2, 4)	144	9	9
(4, 4, 4)	59	6	6
N-D	203	15	15

Table 5.4.: Agreement patterns after the first pass, second pass, and for the final result.

shows the four cases in which the system sentence was the preferred sentence (SY>BL). The bottom part shows the cases with no decision by the workers (N-D). This happened in two cases: (i) one worker preferred the baseline sentence, one worker the system sentence, and one worker did not find any sentence convincing (1, 2, 4) and (ii) all three workers did not find any of the two sentences convincing (4, 4, 4).

In the first pass, the baseline sentence received a total agreement (1, 1, 1) 75 times. In almost twice as many cases (149), at least one worker found the system sentence to be convincing. This shows that the system sentence was not completely unacceptable and the baseline sentence could not be clearly chosen in many cases. However, there are also 154 cases in which one or two workers found neither sentence to be convincing.

The system sentence received a total agreement (2, 2, 2) 247 times, which is the highest number in the SY>BL section of Table 5.4. This suggests that it was easy for workers to agree on the system sentence because it provided the better reasons. In a slightly less number of cases (240) at least one worker preferred the baseline sentence. This is less compared to the BL>SY case. In 312 cases at least one worker found neither sentence convincing. Percentage-wise, this is also less than the total in the BL>SY section of the table. In general, this indicates that it is easier for workers to agree on a system sentence than on a baseline sentence, which means it should contain a better reason.

The 203 N-D sentence pairs were reposted with one assignment per HIT. The new submission was used to break the tie, i.e., it replaced one of the "neither sentence is more convincing" cases in the agreement pattern. In case of the (1, 2, 4) pattern, 68 of the new submissions found the baseline sentence to be more convincing while 67 found the system sentence to be more convincing. In 9 cases the tie could not be broken. The (4, 4, 4) pattern shows very similar results. In 27 cases the workers preferred the baseline sentence and in 26 cases the system sentence. The tie could not be broken in 6 cases. The fact that in the resubmissions workers did not have a strong preference for baseline or system sentence suggests that these cases are rather hard to assess.

The last column shows the final results for all 1,380 documents after updating the first pass results with the second pass results. The baseline sentence was chosen 473 times (34.28%) while our supporting sentence extracted with the system was choosen 892 times (64.64%). The sentence pairs that could not be assessed were reduced from 203 to 15 (1.09%).

Table 5.5 gives a condensed overview of the results of our experiments. After the first pass (row 1 of Table 5.5), workers preferred the system sentence for 57.9% of the reviews and the baseline sentence for 27.39% of the reviews. For 14.71% of the reviews, the scores of the baseline and system sentence were tied, i.e., no decision could be made. These 203 sentence pairs were reposted in the second pass and the results are shown in row 2. As noted above, the workers did not have a strong preference for either baseline or system sentence which is reflected in nearly evenly split results (46.8% choose the baseline sentence and 45.81% preferred the system sentence). For 7.39% sentence pairs the tie could not be broken and they remain undecided.

Experiment	**# Docs**	**BL**	**SY**	**N-D**	**B=S**
MTurk, first pass	1,380	27.39	57.90	14.71	-
MTurk, second pass	203	46.80	45.81	7.39	-
MTurk final	1,380	34.28	**64.64**	1.09	-
MTurk+[B=S]	1,744	27.12	**51.15**	0.86	20.87

Table 5.5.: MTurk evaluation results. Numbers are percentages or counts. BL = baseline, SY = system, N-D = no decision, B=S = same sentence selected by baseline and system.

Row 3 shows the final results for all 1,380 sentence pairs after updating the 14.71% undecided pairs from row 1 with the new ratings from the second pass in row 2. For 64.64% of the reviews the workers preferred the system sentence while only for 34.28% of the reviews the workers preferred the baseline sentence. The results clearly show that the supporting sentence from our system is superior to the baseline sentence which simply chooses the sentence with the strongest sentiment. Only 1.09% of the reviews remain without a decision.

As noted at the beginning of this section, we excluded 364 sentence pairs (20.87% of the test data) from the MTurk experiments because the baseline sentence and system sentence were the same. When adding these reviews to obtain the full set of 1,744 reviews, then the supporting sentence produced by our system is preferred in 51.15% of the cases and the baseline sentence in 27.12%. In effect, the 364 cases where the baseline sentence and the system sentence are the same, contribute to both the baseline and the system results. This is true for the baseline results since the sentence scores well on the traditional sentiment metrics. Similarily, the supporting sentence scores well on our new content keyword metric, making this contribution possible. Consequently, we add these sentences to the results of both the baseline and the system in order to evaluate their performance on the full data set. Thus, for all reviews our system finds a good supporting sentence 72.02% of the time (51.15+20.87) whereas the baseline does so only 47.99% (27.12+20.87) of the time.

5.4.4. Error analysis

In this section we discuss several problems which we identified using an error analysis.

For the evaluation we used relative judgments instead of absolute judgments by judging all possibilities (in our case the baseline sentence and our extracted supporting sentence) in relation to each other. Judging which sentence contains a more convincing reason is a very subjective task. There might be clearer cases like the following example:

(5.11) a. *"The camera is bad."*
b. *"The picture quality is bad."*

Example 5.11a is just a general statement about a camera without a reason why exactly it is bad while example 5.11b gives a supporting fact by being specific about the picture quality. The instructions of our evaluation on MTurk also explained that a sentence like in 5.11a does not contain a reason. In such cases, a sentence like 5.11b should clearly be the preferred sentence.

That being said, not all cases are clear and easy to assess. In fact, a large portion of our system sentences that were rated worse than the baseline sentence did contain a convincing reason. However, the baseline sentence in these cases also contained a reason which was preferred by the turkers. Consider the following examples:

(5.12) (BL) *"The best thing is that everything is just so easily displayed and one doesn't need a manual to start getting the work done."*

(SY) *"The zoom is incredible, the video was so clear that I actually thought of making a 15 min movie."*

(5.13) (BL) *"The colors are horrible, indoor shots are horrible, and too much noise."*

(SY) *"Who cares about 8 mega pixels and 1600 iso when it takes such bad quality pictures."*

In examples 5.12 and 5.13 both the basline sentences (BL) and the system sentences (SY) contain reasons why the camera is good and bad, respectively. If two sentences contain convincing reasons it is not possible to define which of the sentences is objectively better. If both of the presented sentences contained convincing reasons, turkers were asked to choose the sentence which they found more convincing. Since this is a very subjective task, turkers picked a sentence based on personal preferences. This means, while the turkers preferred the baseline sentence in some cases, our system sentence still contained a convincing reason, thus is a valid supporting sentence.

Some errors occurred due to problems in the preprocessing of the text. This is illustrated in example 5.14:

(5.14) (BL) *"Gives peace of mind to have it fit perfectly."*

(SY) *"battery and SD card."*

The system sentence consists of only two noun phrases with the rest of the sentence missing. The reviews that we used for our experiment are user-generated content, which is prone to typographical errors, wrong grammar, and wrong or missing punctuation. We cleaned the dataset as described in Section 5.4.2, however, not every problem could be corrected. As a result, some of the sentences were split incorrectly in the preprocessing step, resulting in incomplete sentences.

To obtain better results, it is necessary to improve cleaning and preprocessing of the user-generated content. This could be, for example, improving preprocessing steps like better sentence boundary detection (e.g., Palmer and Hearst (1997); Clark and Issco (2003); López and Pardo (2015)) and text normalization (e.g., Sproat et al. (2001); Aw et al. (2006); Clark and Araki (2011)).

One of the problems we observed was that in some cases the baseline sentence and the system sentence had a different polarity like in example 5.15:

(5.15) (BL) *"It shares same basic commands and setup, so the learning curve was minimal."*

(SY) *"I was not blown away by the image quality, and as others have mentioned, the flash really is weak."*

The reason for this is that we do not have a dataset with gold labels at the sentence level but use a sentiment classifier to determine the sentiment of each sentence automatically. Since the training input is noisy, the classifier is likely to misclassify some of the sentences. Another reason is that for short reviews with less than five sentences, which have the same polarity as the overall document, we also use sentences with the opposite polarity (that have the lowest confidence score) to obtain a minimum of five sentences per review. We argued that sentences of the opposite polarity and a low confidence score might have the correct polarity, but this is not always the case.

If the two sentences presented in the task were of opposite polarities and both contained reasons, the AMT annotators were often confused. We noticed that in such cases the workers tended to prefer the sentence with the positive polarity, even if the negative sentence contained a more convincing reason than the positive sentence.

Another problem arises due to how the weighting function of our approach is designed: it counts the number of noun phrases and compound noun phrases for every sentence. Sentence with many noun phrases (especially highly frequent nouns) will have a higher score. Thus, the weighting function favors sentences containing many noun phrases. In many cases, having more nouns in a sentence provides a more convincing reason. However, there are some cases where this is not true like in the following example:

(5.16) (BL) *"I have owned my cd300 for about 3 weeks and have already taken 700 plus pictures."*

(SY) *"It has something to do with the lens because the manual says it only happens to the 300 and when I called Sony tech support the guy tried to tell me the battery was faulty and it wasn't."*

In example 5.16 the system sentence is much longer than the baseline sentence and contains more noun phrases, including some highly frequent nouns (e.g., *"lens", "battery"*). While the sentence is about a problem with the camera, it does not explain what exactly the reason is. Moreover, the part about the tech support has nothing to do with the product and the part about the battery is irrelevant since there is no problem with the battery. In this example, the baseline sentence also does not contain a convincing reason, but it was preferred by the workers. The subset of sentences that is used as input for the weighting function sometimes contains another sentence with a better reason, which has not been selected by our system since it has fewer (highly frequent) nouns. This problem is not easy to solve since in many cases a sentence with many (highly frequent) noun phrases does contain a convincing reason. Restricting the sentence length or number of noun phrases would also filter out good supporting sentences.

One of our assumptions was that good supporting facts are most often expressed by noun phrases. In our data we also found sentences without good keyphrases such as the following:

(5.17) *"I have had an occasional problem with the camera not booting up and telling me to turn it off and then on again."*

The sentence in example 5.17 contains two noun phrases (*"problem", "camera"*), which are not keyphrases. Because of the lack of keyphrases the sentence gets a very low overall sentence score by the weighting function. However, this sentence does contain a convincing reason. But instead of noun phrases, the reason is expressed by verbs and particles (*"not booting up and telling me to turn it off and then on again"*).

The weighting function we designed is flexible and can be extended with information about verbs, adjectives, particles and other linguistic expressions. This way, sentences that contain good reasons expressed by linguistic expressions other than nouns can be included.

5.5. Related work

Much research has been done in similar directions as our approach, such as aspect-oriented summarization, pro/con summarization, and approaches that extract summary sentences from reviews. Aspect-oriented summarization collects sentiment assessments for a given set of aspects and returns a list of pros and cons about every aspect. Pro/con summarization lists all sentiment assessments independent from the aspects in pros and cons lists.

One of the earliest contributions addressing aspect-oriented summarization was made in (Hu and Liu, 2004a,b). In the first step, they extract the most frequent aspects of products (e.g., *"picture quality", "size"*) mentioned by users in their product reviews. In the second step, they extract opinion words that can be used to express an opinion about an aspect (e.g., *"good", "horrible", "fast"*). These opinion words are then analyzed to determine the sentiment (positive or negative). To extract infrequent aspects, they analyze all sentences without frequent aspects and if one or more of the extracted opinion words occurs in the sentence, the nearest noun phrase is taken as an infrequent aspect. At the end, for each aspect they found, they list all sentences that contain opinion words as a kind of summary of this aspect.

Kobayashi et al. (2004) extract aspects and opinion words with the use of co-occurrence patterns such as `<Attribute> of <Subject> is <Value>` (e.g., *"the leather seat of Product_X is comfortable"*). They perform experiments on Japanese web documents about cars and video games and compare their semi-automated collection process to a manual collection process.

Popescu and Etzioni (2005) present the framework OPINE, which extracts features (aspects) and opinions from reviews in an unsupervised fashion. Their approach can be broken down into four parts. First, they identify features by extracting noun phrases that have a higher frequency than a threshold they set experimentally, and computing a PMI (pointwise mutual information) score between the noun phrases and meronymy discriminators of classes they identify for their products. Second, they extract opinion phrases by using syntactic dependencies between features and potential opinion phrases. Third, they detect the polarity for each opinion word (positive, negative, or neutral). Finally, they rank opinions based on their strength. Their work is similar to Hu and Liu (2004b)'s approach, however, they improve the steps by assessing candidate features to increase precision, and include composite and implicit features.

Gamon et al. (2005) introduce PULSE, a prototype system that extracts not only sentiment but also topics from reviews and presents them to users in a visual user-interface where they can explore the topics with the associated sentiment. The topics are obtained by clustering the sentences with a keyword-based soft clustering algorithm that uses tf-idf weighting and phrase identification. Since the clustering method does not differentiate between aspects of the product and other keywords, the resulting topics are not necessarily aspects of the product, but can also be terms like *"small"* or *"makes"*.

Zhuang et al. (2006) perform aspect-oriented summarization on the movie domain, using an approach that includes the use of WordNet, statistical analysis, and movie knowledge. Their approach works in a similar fashion to Hu and Liu (2004b)'s approach: they extract features (aspects) and opinion words in each sentence and determine the sentiment of the opinion word. To find features and opinion words they use WordNet, movie casts, and labeled training data. After this, they identify valid feature-opinion pairs and produce a list of positive and negative sentences for each feature.

Blair-Goldensohn et al. (2008) note that previous studies usually do not take a priori knowledge of the domain into account. They aim to improve their aspect-oriented summarization system by leveraging context and include user-provided polarity labels for the review. They further improve results by using a hybrid approach to extract aspects, which is divided into dynamic extraction (extracting aspects from the existing review texts) and static extraction (identifying coarse-grained aspects, labeling training data with these aspects; for example, a sentence *"I loved my meal"* would be annotated with the coarse-grained aspect *"food"*), then using the training data to extract sentences that contain fine-grained aspects corresponding to the coarse-grained ones.

Kim and Hovy (2006) build on previous approaches (Hu and Liu, 2004b; Popescu and

Etzioni, 2005), but instead of extracting features their focus lies on extracting pro and con sentences, which do not necessarily have to contain a feature of the product as long as they explain why the author wrote the review. As training data they use reviews from websites where users can additionally assign pros and cons for the product, and automatically extract triplets of the form `<review text, pros, cons>`. They use the pros and cons provided by the author of the review to label the corresponding sentences in the review, then train a maximum entropy model on the training data to extract pros and cons from unseen reviews.

Qin et al. (2008) refine pro/con summarization by using semantic, syntactic, and lexical features. To better capture the relation between an aspect and a polarity word, they use the syntactic path between the polarity word and its target. Other features include words modifying the polarity word or the path, and lexical information about the word, POS tag, and negation. They use these extracted features to train a sentiment classifier. Their final system classifies the sentences as positive or negative and if more than half of the sentences have been classified as positive, the system recommends the product to the user. Berend (2011) performs a form of pro/con summarization that extracts phrases with reasons instead of whole sentences. The phrases are limited to four tokens at most (e.g. *"small keys"* or *"the screen is tiny"*). They first extract candidate terms with the use of several features (including linguistic, orthographic, world knowledge-based, document and corpus-level features) and train maximum entropy classifiers on them. When running the classifiers on test reviews, the phrases with the highest posteriori probabilities are chosen. The disadvantage of the approach is that besides not differentiating between good and bad reasons, short phrases are also harder to read compared to a well-formed sentence or might not be fully expressive because of the four token limit.

Jin et al. (2016) extract aspects from reviews as well as reasons for why consumers liked or did not like the product. They identify sequences of words that serve as reasons with the help of conditional random fields. However, their extracted reasons are very short and not always clear (e.g., for the aspects of battery they extracted reasons such as *"GPS"*, *"call"*, or *"email"*). Angelidis and Lapata (2018) use neural networks for their opinion summary approach. Their system combines several individual modules (aspect extraction, polarity prediction, and a module for detecting redundant opinions). They also create a new dataset for evaluating opinion summarization systems, which consists of Amazon reviews of different product types.

Aspect-oriented summarization and pro/con summarization are different from our approach that extracts supporting sentences. First, aspects and pros/cons are usually taken from a fixed inventory (e.g., the aspects users can vote on provided by the website, or the pros/cons listed by users). Such an inventory is typically small and does not cover the full spectrum of relevant information. Second, aspect-oriented summarization requires classification of phrases and sentences according to the aspect they belong to. However, not all aspects are explicitly mentioned in review texts. For example, a sentence like *"The camery is very light"* is implicitly referencing to the aspect *"weight"*. To capture implicitly mentioned aspects, systems need to implement some detection tool that recognizes such phrases and assigns them to the corresponding aspect. This is time-consuming and has to be redone for each domain. In addition, correctly determining and matching such phrases with the correct aspect is a hard task and is prone to missing phrases and misclassifying aspects, which results in bad summaries. Our approach is not bound to a fixed inventory and is therefore able to find strong supporting sentences without limits. It also does not involve any manual work of creating an aspect inventory and there are no requirements on the format of reviews such as author-provided pros and cons.

Our approach addresses a use case that requires a short, easy-to-read summary. Aspect lists and pro/con tables are often filled with keywords or phrase snippets, or might not contain any explaining sentences, which is harder to understand than one extracted and well-formed sentence. Our supporting sentence also has the advantage that it conveys the information in the exact manner in which it was provided by the user, not in a shortened way—possibly cut down in the several complex processing steps that might take linguistic material out of the context—that it can be misinterpreted.

Some approaches are concerned with summarizing opinions from documents. Lloret et al. (2012) present a system that combines opinion retrieval, opinion mining, and opinion summarization. After retrieving documents, they perform opinion mining to classify sentences into positive and negative. They then score the relevance of each sentence with a scoring function that takes into account the number of noun phrases (while assuming that longer noun phrases are of more importance) and the frequency of each of the words in the noun phrases. The final summary is composed of the sentences with the highest relevance score. The same relevance score is also used by Raut and Londhe (2014) in their opinion mining and summarization system.

Meng et al. (2012) perform opinion summarization on twitter. They first exploit topic modeling to obtain topics from twitter hashtags. They identify tweets with reasons by using a pattern approach that matches syntax trees but also considers paraphrases of patterns (e.g., not only *"this is why"* but also *"which is why"* and *"that is why"*). The sentiment polarity is detected towards the target of the tweet. They obtain summaries by combining several kinds of information, including the topic, the polarity, as well as what was extracted with their pattern approach.

Only few approaches have attempted to extract single sentences from reviews in the context of summarization. Beineke et al. (2004) present a system that extracts a single passage from a document, which represents a key aspect of what the author thinks about the review subject. They explore this task in the domain of movie reviews and train a classifier on data from the website `rottentomatoes.com`. Rotten Tomatoes provides two types of reviews: written by users of the website and written by "critics"—reviewers who provide high quality reviews and are approved by Rotten Tomatoes. To provide a preview of the review, Rotten Tomatoes displays a summary sentence of the review. These sentences sometimes contain a specific reason for the overall sentiment of the review, but in other cases the sentences do not provide a good reason (e.g., *"It's downright scary how good this movie is."*) and are sometimes just catchy lines with the purpose of drawing moviegoers to read the entire review (e.g., *"El Bulli barely registers a pulse stronger than a book's"*).

Arora et al. (2009) extract sentences from reviews and classify them into two types of sentences: qualified claims and bald claims. A qualified claim gives the reader very specific details about the product. For example, the sentence *"This camera is small enough to fit easily in a coat pocket"* is considered a qualified claim because it explains the characteristics of *"small"* in detail. A bald claim is open to interpretation. For example, the sentence *"This camera is small"* is a bald claim because it is not clear whether *"small"* is positive or negative and depends on the users' needs. Qualified/bald claims differ from assessing the quality of reasons in sentences. Qualified claims do not have to contain a reason (e.g., *"I didn't like the camera, but I suspect it will be a great camera for first timers"*). A bald claim, on the other hand, can contain an informative reason (e.g., *"The size of the camera is small"*).

Our approach of extracting reasons for why customers like something or not shares similarities with the research area of *argument mining*, especially the sub-field of *argumentative sentence detection*. The goal of argument mining is to identify different argument structures in text, such as the premise (also called claim), the conclusions (also called evidence or reasons), as well as the relation or inference between them. An overview of argument mining can be found in the literature (Peldszus and Stede, 2013; Lippi and Torroni, 2016; Stede et al., 2018; Lawrence and Reed, 2019).

Only little research has been done combining argument mining and sentiment analysis. Villalba and Saint-Dizier (2012) examine argument structures and find that arguments are often found in sentences with evaluative expressions (e.g., *"cheap fare"*) or discourse structures such as justifications (e.g., *"because of X"*). Based on their observations they build a rule-based system that extracts arguments from opinionated sentences. Rosenthal and McKeown (2012) propose a system that automatically extracts a certain type of opinionated claims (namely beliefs of the person expressing the claim). They use a supervised machine learning approach that incorporates different features, including sentiment and belief detection. Wachsmuth et al. (2014) build a model of review argumentation and use different features (e.g., discourse relations) to improve their sentiment analysis system. They use argument structures to support polarity classification of their input but do not extract supporting reasons for the classification results. Wachsmuth et al. (2015) claim that in reviews, people use certain sequences to express their opinions and attempt to model this "sentiment flow" (in a general way, not by extracting specific reasons). Dragoni et al. (2018) present their opinion summary system *SMACk*, which combines argument mining with aspect-based opinion mining. They create an argumentation graph, where each node is an argument (represented by extracted triples from text, e.g., `<Aspect,Opinion,Attack>` or `<Aspect,Opinion,Support>`; where opinion is the sentiment polarity of the argument) and edges represent attacks between arguments.

Faulkner (2014) performs an extensive study on argument stance detection. One part of the study is an application for supporting argument summarization, which is inspired by our approach in this chapter. The application takes argumentative student essays as input and extracts a single sentence summary per essay. The goal of the summary is to convey the overall argument stance of an essay. Instead of only taking nouns as in our approach, Faulkner (2014) compiles a lexicon of 108 discourse markers (e.g., *"because"*, *"on the other hand"*, and *"finally"*). For evaluating the supporting arguments, the author also uses a crowdsourcing method and adopts the same scoring system as we did.

Emotion analysis is closely related to sentiment analysis. Ekman (1972) defines six basic emotions (anger, disgust, fear, happiness, sadness, surprise) but the categories can be expanded to include more fine-grained emotions. The task of *emotion detection* or *emotion classification* is concerned with correctly detecting the correct category for a given input text. Analogously to sentiment polarity classification, emotion classification exists on different levels, for example, the document level (e.g., Mishne (2005); Yang et al. (2007b); Kim et al. (2013); Julio and Ayu (2017)) and the sentence level (e.g., Alm et al. (2005); Yang et al. (2007a); Chaffar and Inkpen (2011); Li et al. (2015); Tafreshi and Diab (2018)).

While a large amount of research focuses on classifying emotions of a piece of text into different categories, a closely related task is *emotion cause extraction* (also called *emotion cause detection*). The goal of emotion cause extraction is to identify the reasons for the emotions in text and is therefore similar to our approach. Early work on this task started about a decade ago (e.g., Lee et al. (2010); Chen et al. (2010)) and has gained more interest recently (e.g., Neviarouskaya and Aono (2013); Ghazi et al. (2015); Gui et al. (2016)). In most recent years, researchers used different neural network models for emotion cause detection such as convolutional neural networks (e.g., Chen et al. (2018);

Yu et al. (2019)), long-short term memory networks (e.g., Cheng et al. (2017)), and long-short term memory networks with a multi-attention mechanism (e.g., Li et al. (2019)), to identify causes from text. There exist several corpora for evaluation, annotated with the cause of the respective emotion (e.g., Lee et al. (2010); Gui et al. (2014, 2016); Ghazi et al. (2015); Cheng et al. (2017)).

5.6. Summary

In this chapter we presented a system that extracts *supporting sentences* from reviews. Supporting sentences are single-sentence summaries of reviews that contain the sentiment and a convincing reason for the author's opinion about a product. We used a 2-step approach to obtain supporting sentences. First, a sentiment classifier detects a set of sentences with the strongest sentiment of the same polarity as the review. Second, noun phrases of each sentence are weighted with a weighting function, based on whether they are keyphrases or non-keyphrases, as well as whether they are single nouns or compound nouns. Since no gold standard dataset is available, we used a novel comparative evaluation methodology with the crowdsourcing framework Amazon Mechanical Turk. We showed that our system performs better than a baseline of extracting the sentence with the strongest sentiment.

6. Conclusion

6.1. Summary of contributions

The goal of this thesis was to develop task-oriented specialization techniques for entity retrieval with linguistically informed methods. We investigated three issues which tend to be considered of subordinate concern in standard retrieval approaches, but can improve the user's experience in these specific cases, and proposed solutions to address and resolve these issues with natural language processing methods taken from the research areas named entity disambiguation and sentiment analysis. In this section we summarize the contributions we made in this thesis.

Research Question 1 How helpful is it for named entity disambiguation to expand context in a linguistically informed way by adding the context of expressions from the same document that are (automatically) identified as coreferent?

We investigated this first research question in Chapter 3, in the context of disambiguating named expressions and linking them to their corresponding real world entity. We first proposed *r-context*, a new type of structurally informed context that is created by concatenating sentences that are relevant to the entity. This is in contrast to previous work that uses the entire document as context (e.g., Bollegala et al. (2006); Ikeda et al. (2009); Hoffart et al. (2011)) or a fixed context window around the entity (e.g., Nguyen and Cao (2008); Ikeda et al. (2009); Han et al. (2011); Li et al. (2013)). The relevant sentence selection was done by using coreference chains, which we annotated automatically. For each mention in the coreference chain we extracted the corresponding sentence the mention occurs in, and added it to the relevant context depending on different filter methods.

There was no annotated dataset for our task. Thus, we decided to create a pseudo-ambiguity dataset, similar to previously used pseudo-ambiguity datasets for word sense disambiguation (Schütze, 1992; Gale et al., 1992), which allowed us to perform a systematical study. For comparison reasons, we created a small dataset of real data, where we manually labeled the person ambiguity information.

We experimented with different corpus sizes, corpus creation methods, and filter methods to obtain different types of contexts (i.e., selecting sentences depending on the coreference information). We showed that our approach of using r-context improved disambiguation results in most cases, compared to using the entire document or a fixed window around the entity.

Research Question 2 How can we learn general properties that allow us to identify a little-known entity without having seen the entity in context before?

In Chapter 4 we continued investigating methods for improved named entity disambiguation and proposed a system that improves identification for less known entities, which have often been neglected in previous work. The underlying idea of the system is that even if we have no or only little training material for a specific entity, we still know some properties (e.g., profession and nationality) about the entity from name authority files such as OCLC's WorldCat Identities or the Virtual International Authority File. Using this information, we can create an aggregate of textual material about different people with the same properties, which can be used as a proxy for disambiguation of less known entities with these properties. The disambiguation was done with a topic modeling approach that obtains topic information from the textual material as well as from the context around the entity that needs to be disambiguated, and then compares the topic information using cosine simularity to find the most probable target entity.

We experimented with different parameters (context size around the proper name, number of topics, different corpora) to evaluate how the prediction quality is affected. Our results showed that our approach is very helpful for disambiguating (i) entities for which not much training material is available, and (ii) entities with little surrounding context (such as a single sentence only), both of which are useful for many applications.

Research Question 3 How can opinions about entities be presented in an effective way so that users can see not only the sentiment (positive/negative) but also a reason for the sentiment?

We addressed the final research question in Chapter 5, in the context of user-generated product reviews and with methods from the NLP research area sentiment analysis. We first introduced the concept of a *supporting sentence*, a sentence that supports the author's assessment about whether a product is good or bad together with a convincing reason. We then presented a system which extracts supporting sentences from reviews in two steps. In the first step, it applies a sentence level sentiment classifier on reviews to identify the sentiment of each sentence (positive or negative). In the second step, the system applies a special weighting algorithm to the set of sentences with the strongest sentiment (with the same polarity as the entire review) and chooses the sentence with the most convincing reason. We designed the weighting algorithm based on the observations that supporting facts are most often expressed by noun phrases, and that domain-specific noun phrases and compound nouns provide more information.

To evaluate our system we defined a new evaluation methodology based on relative judgments, and designed the task in such a way that it asked annotators to decide, given two sentences A and B, which one has the more convincing reason. For the annotation we explored Amazon Mechanical Turk, a crowdsourcing service where requesters can post tasks that require human workers to solve them. We demonstrated that a crowdsourcing service can be used to obtain a large amount of annotations without the need to hire experts, and discussed design questions and quality control measures for using such a service. We showed that in many cases our extracted supporting sentences contain more convincing reasons than the sentences with the strongest sentiment.

6.2. Final remarks

Our contributions in the context of the recent development in natural language processing

The methods and models used for natural language processing and machine learning in general have changed considerably in the last decades. Early approaches were generally rule-based systems where rules and constraints had to be designed by hand. In the 1990s, large annotated corpora became available, largely because of the Message Understanding Conference (Grishman and Sundheim, 1996) and later the Automatic Content Extraction program (Doddington et al., 2004). With the advent of these and other annotated corpora, approaches slowly shifted from rule-based systems to data-driven machine learning models.

Another change has occurred in natural language processing during the last decade with the increasing popularity of a special type of machine learning that uses artificial neural networks: deep learning. Early models of the various neural network architectures that researchers use nowadays were already introduced in the last century, for example, the origin of the convolutional neural network, called neocognitron (Fukushima, 1980), recurrent neural networks (Rumelhart et al., 1986), and long short-term memory networks (Hochreiter and Schmidhuber, 1997). While artificial neural networks had been used for a while already, deep neural networks (where the "deep" refers to the network consisting of multiple hidden layers) only started becoming popular in the 2000s due to the availability of more computing power that is necessary to process data through a deep network structure.

Neural networks have been used for various tasks in natural language processing, for example, speech recognition (e.g., Graves et al. (2013)) or machine translation (e.g., Bahdanau et al. (2015)), and have proven to work well and outperform state-of-the-art approaches in many situations. Deep learning with neural networks has also been applied to the research areas we worked on in this thesis: named entity disambiguation and entity linking (e.g., He et al. (2013); Sun et al. (2015); Francis-Landau et al. (2016); Raiman and Raiman (2018)) and sentiment analysis (e.g., Socher et al. (2011); dos Santos and Gatti (2014); Severyn and Moschitti (2015); Xue and Li (2018)). With the rapid improvements from neural network approaches, the question arises why we did not use neural networks in our work. We have partly touched on this question in some of our related work sections but now give a more detailed answer in this section.

Neural network approaches perform very well on many standard tasks. In the research area of named entity disambiguation, a classic example is linking an entity to its Wikipedia article. For sentiment analysis, an example is polarity classification (of different granularity). For well-established standard tasks like these, there usually exist large training corpora that allow neural networks to learn extensively and thus enable them to perform with great success. However, neural networks are not the best choice in every situation as there are still limitations.

The core part of our approach in Chapter 3 shows that linguistically motivated context can improve named entity disambiguation. For our experiments we chose three well-established statistical classifiers. It was important to us to show the effectiveness of our context with relatively "simple" methods to show that it is indeed the context choice that improves the results, not a complex model such as a neural network. Obviously, it is possible to use our relevant context as input in deep learning models as well. Neural network approaches often take the entire document as input and let the network decide which parts contain the most important information. However, it was shown that taking the entire document does not always provide best results even with neural networks and that a better context choice can be beneficial (Sil et al., 2017). This shows that the model choice of a neural network and tuning its parameters is not necessarily enough to produce good results and that a linguistically motivated input structure can be important for improvements.

In this thesis, we focused on two special groups of named entities. In Chapter 3 we evaluated our approach on a dataset with highly ambiguous entities and in Chapter 4 we proposed a system that identifies little-known entities. In both cases, there is usually not much training material available. Many systems in the literature are evaluated on standard datasets. However, it was found that most datasets contain little ambiguity and are biased towards well-known entities for which more training data is available (van Erp et al., 2016; Ilievski et al., 2016). This results in "semantic overfitting" and systems performing better on such datasets than on datasets that contain more ambiguity and less known entities (Ilievski et al., 2018). A general property of neural networks is that they require large amounts of data to learn enough information. Therefore, they work well on these standard datasets but struggle in a setting where little-known referents with almost no training data need to be disambiguated. Thus, for specific tasks with little training material it still remains a challenge to formulate the problem in a linguistically motivated way and to find the appropriate system to solve it. Our contribution in Chapter 4 provides a solution that can deal with such a problem, which is still problematic for neural networks.

In Chapter 5 we extracted single supporting sentences from reviews, which is a non-standard task in the research area of sentiment analysis and for which no training corpora are available. It is still an open question whether neural networks would be beneficial for this task or not. In emotion cause detection, which is a similar task to ours, Cheng et al. (2017) compared an LSTM model with an SVM, based on a corpus created from tweet data, and found that the LSTM model performed significantly worse than the SVM. They suspect that their LSTM model was not able to learn sufficiently because of a lack of grammatical information in the user-generated content. More promising results were achieved in later work (Chen et al., 2018; Yu et al., 2019) but there is still much room for improvement, which shows that the task is still hard to solve for neural networks.

To summarize, there is no doubt that neural networks have advanced greatly especially on standard tasks where large amounts of training data are available. However, we believe that in certain situations neural networks are not always the best choice, or can benefit from linguistically motivated input. Such situations can be, for example, when not enough training material is available and obtaining sufficient data might be hard or infeasible. The solutions we provide in this thesis can be used in such situations as they do not require large amounts of training data and can therefore bridge an important gap.

Further refinements

Our approaches in this thesis are for specific sub-tasks and can be refined, expanded, and integrated into larger systems and applications, such as search engines (in general, for people searches, or for product searches) or question answering systems. In this section we will describe some possible refinements that can be done.

Informed context selection. In Chapter 3 we used coreference information to obtain sentences that are relevant to an entity. Since gold annotations are usually not available in a real scenario, we applied a coreference resolution tool to our data to predict the information automatically. As discussed in the error section of Chapter 3, some problems were caused by the automatic detection of coreference. To avoid missing relevant sentences or adding irrelevant sentences to the context, it is important to obtain a more reliable coreference annotation from an automatic tool. A possibility to improve the results is to use a combination of coreference resolvers ("model stacking", e.g., Björkelund and Farkas (2012); Clark and Manning (2015)).

We selected sentences to be included in the relevant context if the entity is mentioned in the sentence. This selection process can be refined further. Not every sentence contains information about a single entity only. In some cases, a sentence contains information about two or more entities. Further linguistic processing can determine which parts of the sentence are really about the entity in question and to what extent. Then, it is possible to decide to (i) take the entire sentence, (ii) take only parts of the sentence, or (iii) discard the sentence. This linguistic analysis can include, for example, determining the predicates and semantic arguments of the sentence with a semantic role labeler (e.g., Gildea and Jurafsky (2002)) to learn more about the entity in question. By doing this, less noise will be added to the relevant context.

We also saw that information about an entity changes over time. For example, two people with the same name could be sportsmen on the same team but during different time periods. Depending on the date of a document, older information might be more useful for the disambiguation of a person than newer information and vice versa. Thus, an interesting extension would be to incorporate time information of documents to obtain context which is relevant given specific time periods.

Identification of little known referents. In our pilot study in Chapter 4, we focused on two properties of people: nationality and profession. However, our system can be extended to include more properties. These properties can include, for example, date of birth, date of death, works (songs, movies, books, etc.), or related entities. The OCLC's WorldCat Identities website also provides a section called "associated subjects" for entities. The more associated a topic is with the entity, the larger it is visually presented. It might be possible to exploit these subjects to identify entities, although some filtering will be necessary as some of the associated subjects are not prominent properties of the entity (e.g., *"actors"* for *"Michael Jackson, the American singer"*[61]).

Our evaluation was done on a fixed set of entities for which we created a fixed set of corpora with the properties corresponding to the entities. To make the system more flexible for a large pool of entities, it is better to create individual corpora for each property (e.g., an *"American corpus"* and a *"singer corpus"*) and then combine these corpora. Properties corpora for a large amount of entities can be created automatically for the most part, for example, by using extended lists of possible nationalities, professions, and other properties.

Informative sentence summaries. The system that we presented in Chapter 5 already performs the core part of the task: extracting a supporting sentence from a review (or any other document). This supporting sentence can be integrated into larger systems. One scenario is to visualize supporting sentences similar to results from a search engine as we mentioned in Chapter 5, Figure 5.3, where the query is the product of interest. This way, users and companies can quickly skim through all the supporting sentences and then decide which reviews they want to read in more detail.

Another visualization application is to cluster the extracted supporting sentences and then present them to users by aspects and reasons. This way, users can easily find information about specific aspects of a product and explore them in more detail with the help of the associated reasons.

Our weighting function was limited to keyphrases which consisted of simple nouns and compound nouns. This weighting function can be extended to integrate weights for other constructions that express good reasons, such as verbal constructions.

[61]`http://www.worldcat.org/identities/lccn-n83133203/`

A. Appendix to Chapter 3

The results for our approach of using relevant context to disambiguate proper names presented in this thesis differ from the results presented in our previous publication (Glaser and Kuhn, 2014). The reason for this is that the algorithm that distributes samples into folds for the cross-validation has been changed after the publication. For more information about what was changed we refer to Chapter 3, Section 3.4.3. All results in Chapter 3, Section 3.4.4 as well as the following tables in this Appendix are based on the new version of the algorithm that distributes samples randomly into folds (instead of equally).

The results for each experiment have changed slightly due to the different distribution of samples in the folds. However, the main trends of the results described in Chapter 3, Section 3.4.4 are the same after changing the algorithm:

- Using more context improves the results (the 11 baseline performs better than the 00 baseline).
- Our approach of using relevant context outperforms both baselines in most cases.
- There is no clear pattern of which type of relevant context is best.

			Baselines		Filter Methods			
Class.	Y	Docs	00	11	NNP	NNP+NN	NNP+PRP	NNP+NN+PRP
NB	2	100	77.5	**87.9**	78.9	78.9	80.1	79.8
		200	72.8	80.0	79.9	79.9	78.3	**80.5**
		300	68.6	76.4	83.2	**83.6**	82.5*	83.1
	3	100	77.3	**87.5**	85.0*	85.0*	81.4*	81.4
		200	76.0	84.4	**85.3**	84.7	85.0*	84.8
		300	70.3	77.9	**88.6**	**88.6**	88.3	87.8
	4	100	81.6	90.0	93.0	93.0	**93.7**	**93.7**
		200	82.0	90.0	94.0	**94.2**	92.3	92.3
		300	80.3	89.7	**93.8**	93.4*	92.4	92.3
J48	2	100	70.7	83.2	**93.8**	**93.8**	90.0	90.0
		200	66.0	78.8	85.4	**87.6**	84.7	84.7
		300	69.3	76.7	81.0*	80.5	**82.8**	81.4
	3	100	68.0	83.2	91.6	91.6	**91.8**	**91.8**
		200	71.9	82.1	87.7*	**89.1**	88.9	**89.1**
		300	71.3	78.0	76.0	77.5	78.0	**80.0**
	4	100	76.0	86.7	**94.3**	**94.3**	87.7	93.4
		200	78.9	87.5	92.4	92.4	92.2	**92.5**
		300	79.2	88.4	95.7	**95.9**	93.2	92.6
SMO	2	100	77.1	87.7	**96.9**	**96.9**	95.1*	96.5*
		200	73.5	81.5	94.4	94.4*	95.2*	**95.4***
		300	73.7	79.0	**87.7**	**87.7**	86.9*	86.9
	3	100	77.9	87.4	**95.9**	**95.9**	95.8	95.8
		200	76.8	84.5	95.7	95.4	96.2	**96.4**
		300	75.4	80.1	89.9	90.7	91.3	**91.4***
	4	100	84.7	90.6	**98.8**	**98.8**	98.1	98.1
		200	85.6	91.2	97.2	97.2	**97.4**	97.1
		300	86.7	91.8	99.1*	**99.4**	98.8	99.1

Table A.1.: Detailed results of the experiments with relevant context for the entity pair *"Gore/Cheney"*, using three different classifiers (Naive Bayes (NB), J48, SMO) with different parameters for documents (100, 200, 300) and extraction ($Y \in \{2, 3, 4\}$). All results report the weighted micro-averaged F_1 score over both entities in each entity pair. The best result in each row (i.e., parameter setting) is bold. Results of the filter methods are marked with * if the difference to the 11 baseline is statistically significant at $p < .05$.

			Baselines		Filter Methods			
Class.	Y	Docs	00	11	NNP	NNP+NN	NNP+PRP	NNP+NN+PRP
NB	2	100	65.1	**73.0**	71.0	71.9	70.8	70.8*
		200	68.8	76.3	**80.4**	79.4	78.7	78.7*
		300	68.3	76.1	82.7	**83.4**	82.0	81.6
	3	100	66.6	75.7	87.1*	**88.5**	86.4*	85.7
		200	68.5	77.1	86.4	**87.0**	86.9	86.6
		300	71.2	79.3	**92.0**	91.9	91.6	91.7
	4	100	77.1	86.4	96.2	96.3	**96.8**	96.2
		200	68.0	78.1	91.4*	**91.6***	90.0*	90.1
		300	67.9	77.1	88.8*	**89.2**	88.5	88.6
J48	2	100	58.8	65.7	74.0*	**77.0***	69.3	64.7
		200	65.9	71.5	80.2	75.1	**80.3**	75.8
		300	66.4	74.0	77.8	**79.6***	74.1	74.0*
	3	100	64.4	70.2	72.4	73.2*	**74.4**	70.6
		200	68.7	77.0	76.7	**78.4**	76.9*	76.6
		300	70.4	78.5	79.8*	**83.2**	80.2	80.2
	4	100	72.8	80.6	86.5	88.2*	**88.5**	88.3
		200	71.9	79.0	86.5	**87.3**	82.9	83.1
		300	71.5	80.2	82.3*	**83.1***	80.3	79.1
SMO	2	100	68.9	70.9	87.2	**88.5**	84.6	85.0
		200	69.9	75.4	85.3	85.7	**86.0**	85.6
		300	71.1	74.5	84.5	**85.7**	**85.7**	84.9
	3	100	70.4	76.5	87.8	88.2	92.1	**92.3**
		200	72.3	78.4	87.3*	86.6	**90.1**	89.9
		300	73.5	77.7	90.0	90.2	**90.6***	89.8
	4	100	80.3	86.8	93.2	93.6	**94.1**	**94.1**
		200	78.7	82.1	93.8*	93.5	**94.8***	94.5
		300	77.9	82.5	**93.7***	93.0	93.3*	93.5*

Table A.2.: Detailed results of the experiments with relevant context for the entity pair *"Barak/Sharon"*, using three different classifiers (Naive Bayes (NB), J48, SMO) with different parameters for documents (100, 200, 300) and extraction ($Y \in \{2,3,4\}$). All results report the weighted micro-averaged F_1 score over both entities in each entity pair. The best result in each row (i.e., parameter setting) is bold. Results of the filter methods are marked with * if the difference to the 11 baseline is statistically significant at $p < .05$.

			Baselines		Filter Methods			
Class.	Y	Docs	00	11	NNP	NNP+NN	NNP+PRP	NNP+NN+PRP
NB	2	100	80.2	84.8	91.2*	91.2*	**92.6**	**92.6**
		200	75.8	80.0	88.5	88.6	88.7*	**89.5***
		300	76.0	80.5	88.6	88.5	88.8	**88.9**
	3	100	73.3	81.3	**92.7**	**92.7**	90.3*	90.1
		200	74.5	78.7	87.7	**87.9**	87.8	**87.9**
		300	76.3	81.6	**89.9**	**89.9**	89.6	89.7
	4	100	82.3	90.6	**95.9***	**95.9**	95.5*	95.8*
		200	81.4	90.0	96.3	96.3*	96.5	**96.8**
		300	77.2	84.8	94.8*	94.8*	94.9	**95.5**
J48	2	100	71.1	79.3	**85.7**	**85.7**	85.3	85.3
		200	72.1	76.8	**81.7**	**81.7***	**81.7**	81.1
		300	71.5	78.9	81.6	82.5	83.9	**84.4**
	3	100	69.0	76.2	85.1	83.8	**85.6**	81.0*
		200	70.4	75.0	81.9	82.4*	80.0*	**84.9***
		300	75.4	80.0	81.5*	81.6*	80.3*	**84.7***
	4	100	78.9	84.8	**94.3**	**94.3**	90.5	90.5
		200	79.9	87.0	89.8	90.1	**91.4***	91.1
		300	79.3	85.2	85.4	**86.3**	85.7	**86.3**
SMO	2	100	81.1	84.6	91.6*	91.8*	**92.7**	**92.7**
		200	78.2	81.8	88.6	88.6	90.6	**90.7***
		300	78.4	83.8	89.8	89.9	**90.0**	**90.0**
	3	100	76.9	84.7	95.8	**96.1**	95.1	95.1
		200	77.7	83.1	90.2	90.2	**92.8**	92.0
		300	80.5	85.8	94.1	94.0	**94.8***	94.4
	4	100	84.0	91.1	97.2	97.0	**97.5**	97.0
		200	83.7	91.7	**95.7**	**95.7**	95.6	95.6
		300	83.9	89.2	97.1*	97.2	**97.6**	**97.6**

Table A.3.: Detailed results of the experiments with relevant context for the entity pair *"Milosevic/Kostunica"*, using three different classifiers (Naive Bayes (NB), J48, SMO) with different parameters for documents (100, 200, 300) and extraction ($Y \in \{2, 3, 4\}$). All results report the weighted micro-averaged F_1 score over both entities in each entity pair. The best result in each row (i.e., parameter setting) is bold. Results of the filter methods are marked with * if the difference to the 11 baseline is statistically significant at $p < .05$.

			Baselines		Filter Methods			
Class.	Y	Docs	00	11	NNP	NNP+NN	NNP+PRP	NNP+NN+PRP
NB	2	100	69.2	**79.7**	75.0*	76.4	75.1	75.5
		200	73.5	81.4	81.6	78.2	**82.1**	81.9
		300	74.7	83.8	**88.7***	88.2	87.9	87.7*
	3	100	69.2	78.5	**85.6**	85.1	82.1	82.3
		200	74.0	82.0	**94.3**	93.9	91.7	91.9
		300	75.3	83.3	**94.6**	94.1	94.4	94.5
	4	100	71.4	80.3	87.0	**87.4***	86.2	84.6*
		200	73.7	81.1	94.0	93.8*	94.0	**94.8***
		300	75.3	84.0	**95.6**	95.1	95.1*	95.3
J48	2	100	63.4	78.3	88.8	88.1	**88.9**	88.8
		200	67.5	77.9	80.7*	82.2	86.2	**87.2***
		300	70.6	80.2	89.7	**90.4**	88.3	89.4*
	3	100	62.3	74.5	87.0	**87.4**	82.9	85.1
		200	70.9	80.4	90.9*	**91.5***	88.7*	90.3
		300	73.8	83.0	90.6	91.4	91.6*	**92.0**
	4	100	67.1	76.8	88.5*	88.5*	**90.7**	90.5
		200	72.6	82.0	90.9	90.9	**91.6**	91.2*
		300	75.5	84.5	90.7	**91.0**	88.2	89.4
SMO	2	100	71.4	80.5	89.8	88.1	86.5*	**90.2**
		200	73.0	81.2	89.3	89.6*	89.7	**91.2***
		300	75.3	83.5	91.5*	92.0*	89.9	**92.7**
	3	100	69.8	78.1	88.8*	**89.0***	86.2	88.5
		200	74.0	83.9	93.6	94.1	93.5	**94.6**
		300	78.1	85.7	95.3	95.4	95.2*	**95.6**
	4	100	72.4	81.0	**93.6***	92.7*	92.7	93.4*
		200	77.8	85.1	96.1	96.3*	**97.1**	96.9
		300	78.9	87.6	95.3	**96.2**	94.7	95.4

Table A.4.: Detailed results of the experiments with relevant context for the entity pair *"Martin/Faulk"*, using three different classifiers (Naive Bayes (NB), J48, SMO) with different parameters for documents (100, 200, 300) and extraction ($Y \in \{2, 3, 4\}$). All results report the weighted micro-averaged F_1 score over both entities in each entity pair. The best result in each row (i.e., parameter setting) is bold. Results of the filter methods are marked with * if the difference to the 11 baseline is statistically significant at $p < .05$.

			Baselines		Filter Methods			
Class.	Y	Docs	00	11	NNP	NNP+NN	NNP+PRP	NNP+NN+PRP
NB	2	100	82.5	**91.5**	87.5*	87.5	83.7*	83.7*
		200	82.0	92.2	95.4	**96.0***	91.0	90.9
		300	80.3	91.0	**96.5**	**96.5**	92.9	92.3
	3	100	81.9	91.2	**95.1**	94.7	91.4	90.7
		200	80.9	90.5	**99.1**	**99.1***	95.6	95.0
		300	79.6	89.4	**97.5**	97.4	95.5*	94.8
	4	100	82.2	91.7	**95.8**	95.5	93.9	93.9
		200	79.6	89.7	97.4	**97.7**	94.1*	94.1*
		300	79.3	89.4	97.7*	97.7	**97.9**	97.7*
J48	2	100	73.4	88.3	93.6*	93.6*	**96.3**	**96.3**
		200	76.1	88.1	96.1	95.9	96.7	**97.1**
		300	78.9	88.3	**95.4**	94.5	93.9*	**95.4**
	3	100	70.6	87.2	**98.1**	**98.1**	97.9	97.9
		200	76.8	88.8	94.3	94.8	97.2	**97.9***
		300	76.8	88.2	94.6	94.5	**96.1**	**96.1**
	4	100	73.5	86.1	95.0*	95.0	**95.3**	**95.3**
		200	76.1	86.6	94.7*	94.9*	96.5	**96.7**
		300	77.3	88.1	91.1*	91.9	**92.1**	92.0
SMO	2	100	84.7	93.3	98.0	98.0	**98.3**	**98.3**
		200	86.5	93.8	98.4	98.4*	**98.7***	98.5*
		300	86.6	92.6	97.9	**98.3***	98.1*	98.1
	3	100	83.5	93.3	98.9*	98.9*	**99.0**	**99.0**
		200	85.7	92.7	98.7	98.9	98.9	**99.3**
		300	84.6	90.9	98.4	**98.7**	98.3	98.5
	4	100	84.5	92.8	**99.2**	**99.2**	**99.2**	**99.2**
		200	83.6	90.7	98.1	98.1	98.0	**98.3**
		300	84.1	91.9	98.7	**98.9**	98.5	98.5

Table A.5.: Detailed results of the experiments with relevant context for the entity pair *"Manning/Montana"*, using three different classifiers (Naive Bayes (NB), J48, SMO) with different parameters for documents (100, 200, 300) and extraction ($Y \in \{2, 3, 4\}$). All results report the weighted micro-averaged F_1 score over both entities in each entity pair. The best result in each row (i.e., parameter setting) is bold. Results of the filter methods are marked with * if the difference to the 11 baseline is statistically significant at $p < .05$.

B. Appendix to Chapter 4

Table B.1.: Real world entities for each name and number of extracted snippets (#Ex) from Wikipedia, used in the silver standard corpus. Numbers are counts.

Name	Entity	#Ex	Entity	#Ex
David Mitchell	British comedian	111	Scottish hockey player	3
	British writer	57	American figure skater	2
	American martial artist	11	Scottish admiral	2
	British politician	10	Tasmanian lawyer	2
	Australian builder	3		
David Thomas	American singer	47	American politician	1
	American Am. football player	19	Trinidadian beachvolleyball player	1
	British industrialist	8	Welsh poet	1
	British singer	5	Welsh priest	1
	British politician	2		
Jack Johnson	American musician	230	American Am. football player	1
	American boxer	176	American politician	1
	American icehockey player	16	Canadian politician	1
	American actor	1	English footballer	1
John Edwards	American politician	400	Australian TV producer	3
	American racing driver	7	English guitarist	3
	English priest	4	American basketball player	1
John Smith	British politician (1)	107	English Politician	3
	English explorer	93	English cricketer (1)	3
	American actor	75	English poet	2
	British writer	32	English botanist	2
	Scottish architect	30	English brewer	2
	American athlete	19	English writer	2
	American politician	14	English musician	2
	NZ rugby league player	13	Canadian politician	1

Continued on next page

Table B.1: (Continued)

Name	Entity	#Ex	Entity	#Ex
	American wrestler	12	Canadian poet	1
	Am. football player	11	British astronomer	1
	Welsh politician	5	English philosopher	1
	American lawyer	5	American politician	1
	British politician (2)	4	English cricketer (2)	1
	American missionary	3	English cricketer (3)	1
	English engraver	3	Australian politician	1
	English TV executive	3	Australian politician	1
	British politician (3)	3	Australian academic	1
	American colonial president	3	South African rower	1
	American Mormon leader	3	Sergeant	1
John Williams	American composer	650	American Navy sailor	2
	Australian guitarist	83	Welsh sergeant	2
	English missionary	20	Canadian politician	2
	English actor	18	Australian politician (1)	2
	American politician	17	American Am. football player	2
	Welsh bishop	12	Welsh convict	2
	American archer	6	English bishop	1
	American radio commentator	5	Welsh politician	1
	American bishop	4	Australian politician (2)	1
	American equestrian	4	American winemaker	1
	Australianrugby league player	3	English satirist	1
	American basketball player	3	Australian rules footballer	1
	English minister	3	Welsh snooker referee	1
	Welsh lawyer	3	British Army officer	1
	Welsh priest	2		
Michael Collins	Irish leader	364	Irish politician	2
	American astronaut	67	Irish actor	1
	American baseball player	6	Irish writer	1
	American writer	2		
Michael Jackson	American singer	3899	English footballer	4
	British writer	27	British TV executive	3
	American radio commentator	15	English rugby league player	2
	American wide receiver	5	Niuean journalist	2
	Irish bishop	5	American basketball player	1

Continued on next page

Table B.1: (Continued)

Name	Entity	#Ex	Entity	#Ex
	Canadian actor	4	New Zealand anthropologist	1
Michael Moore	American film director	532	(Unknown) herbalist	5
	Australian politician	14	British politician	2
	American musician	7	Scottish footballer	1
	American bassist	5		
Paul Williams	American songwriter	149	American director	5
	American singer	74	English philosopher	4
	American journalist	48	British politician	3
	American architect	39	NZ rugby union player	3
	American boxer	30	Canadian athlete	2
	American musician	16	English bishop	1
	Australian rules footballer	11		
Peter Müller	Swiss skier	10	German politician	5
Richard Burton	Welsh actor	592	American actor	1
	British comic editor	12	British journalist	1
Roger Taylor	British tennis player	39	English musician (2)	17
	English musician (1)	23		
Tony Martin	German bicycle racer	91	Canadian politician	6
	Australian comedian	68	Australian rugby league player	4
	American composer	35	Trinidadian professor	4
	British singer	34	American artist	3
	American Am. football player	9	Australian actor	2
	American singer	9	British darts player	2
	English farmer	8		

Table B.2.: Detailed results of the experiments with properties for the proper name *"David Mitchell"*, with the following parameters: topics=1000, corpus=W$_{all}$, context size=paragraphs. BM = Baseline Majority, BJ = Baseline Jaccard, TM = Topic model approach. The table shows the number of context snippets per entity (#Ex), the number of true positives (TP), false positives (FP), false negatives (FN), and true negatives (TN), as well as the results for precision (P), recall (R) and F$_1$ score. Numbers are counts or percentages.

	Label	#Ex	TP	FP	FN	TN	P	R	F$_1$
BM	Comedian	111	111	90	0	0	55.2	100.0	71.2
	Author	57	0	0	57	144	0.0	0.0	0.0
	Fighter	11	0	0	11	190	0.0	0.0	0.0
	Politician	10	0	0	10	191	0.0	0.0	0.0
	Builder	3	0	0	3	198	0.0	0.0	0.0
	Field hockeyplayer	3	0	0	3	198	0.0	0.0	0.0
	Royal Navy officer	2	0	0	3	199	0.0	0.0	0.0
	Figure skater	2	0	0	3	199	0.0	0.0	0.0
	Lawer	2	0	0	3	199	0.0	0.0	0.0
BJ	Comedian	111	22	40	89	50	35.5	19.8	25.4
	Author	57	28	103	29	41	21.4	49.1	29.8
	Fighter	11	0	8	11	182	0.0	0.0	0.0
	Politician	10	0	0	10	191	0.0	0.0	0.0
	Builder	3	0	0	3	198	0.0	0.0	0.0
	Field hockeyplayer	3	0	0	3	198	0.0	0.0	0.0
	Royal Navy officer	2	0	0	3	199	0.0	0.0	0.0
	Figure skater	2	0	0	3	199	0.0	0.0	0.0
	Lawer	2	0	0	3	199	0.0	0.0	0.0
TM	Comedian	111	97	10	14	80	90.7	87.4	89.0
	Author	57	36	4	21	140	90.0	63.2	74.2
	Fighter	11	11	0	0	190	100.0	100.0	100.0
	Politician	10	5	16	5	175	23.8	50.0	32.3
	Builder	3	3	10	0	188	23.1	100.0	37.5
	Field hockeyplayer	3	0	1	3	197	0.0	0.0	0.0
	Royal Navy officer	2	2	0	0	199	100.0	100.0	100.0
	Figure skater	2	2	0	0	199	100.0	100.0	100.0
	Lawer	2	1	3	1	196	25.0	50.0	33.3

Table B.3.: Detailed results of the experiments with properties for the proper name *"David Mitchell"*, with the following parameters: topics=100, corpus=W_{lp}, context size=paragraphs. BM = Baseline Majority, BJ = Baseline Jaccard, TM = Topic model approach. The table shows the number of context snippets per entity (#Ex), the number of true positives (TP), false positives (FP), false negatives (FN), and true negatives (TN), as well as the results for precision (P), recall (R) and F_1 score. Numbers are counts or percentages.

	Label	#Ex	TP	FP	FN	TN	P	R	F_1
BM	Comedian	111	111	90	0	0	55.2	100.0	71.2
	Author	57	0	0	57	144	0.0	0.0	0.0
	Fighter	11	0	0	11	190	0.0	0.0	0.0
	Politician	10	0	0	10	191	0.0	0.0	0.0
	Builder	3	0	0	3	198	0.0	0.0	0.0
	Field hockeyplayer	3	0	0	3	198	0.0	0.0	0.0
	Royal Navy officer	2	0	0	3	199	0.0	0.0	0.0
	Figure skater	2	0	0	3	199	0.0	0.0	0.0
	Lawer	2	0	0	3	199	0.0	0.0	0.0
BJ	Comedian	111	22	40	89	50	35.5	19.8	25.4
	Author	57	28	103	29	41	21.4	49.1	29.8
	Fighter	11	0	8	11	182	0.0	0.0	0.0
	Politician	10	0	0	10	191	0.0	0.0	0.0
	Builder	3	0	0	3	198	0.0	0.0	0.0
	Field hockeyplayer	3	0	0	3	198	0.0	0.0	0.0
	Royal Navy officer	2	0	0	3	199	0.0	0.0	0.0
	Figure skater	2	0	0	3	199	0.0	0.0	0.0
	Lawer	2	0	0	3	199	0.0	0.0	0.0
TM	Comedian	111	97	10	14	80	90.7	87.4	89.0
	Author	57	36	4	21	140	90.0	63.2	74.2
	Fighter	11	11	0	0	190	100.0	100.0	100.0
	Politician	10	5	16	5	175	23.8	50.0	32.3
	Builder	3	3	10	0	188	23.1	100.0	37.5
	Field hockeyplayer	3	0	1	3	197	0.0	0.0	0.0
	Royal Navy officer	2	2	0	0	199	100.0	100.0	100.0
	Figure skater	2	2	0	0	199	100.0	100.0	100.0
	Lawer	2	1	3	1	196	25.0	50.0	33.3

C. Appendix to Chapter 5

Table C.1.: Top 50 nouns and compound nouns when computing absolute (raw) frequency over the Amazon data.

Count	Noun	Count	Compound Noun
65175	camera	2035	picture quality
25583	picture	1930	battery life
13559	battery	1894	memory card
9987	quality	982	image quality
8246	photo	835	shutter speed
7195	card	684	memory stick
7144	time	673	movie mode
6932	feature	507	film camera
6687	shot	493	lens cap
6384	image	472	price range
6290	mode	427	quality picture
6185	zoom	426	card reader
5694	lens	397	megapixel camera
5206	flash	363	auto mode
4973	price	350	zoom lens
4895	problem	335	sd card
4252	memory	305	image stabilization
4016	software	280	flash card
3975	thing	279	macro mode
3808	setting	266	lens cover
3701	lot	260	battery charger
3572	size	255	customer service
3287	screen	255	shutter button
3101	review	252	view finder
3016	year	236	action shot
2973	computer	229	shutter lag
2954	resolution	226	photo quality

Continued on next page

Table C.1: (Continued)

Count	Noun	Count	Compound Noun
2591	day	224	auto focus
2551	color	211	lithium battery
2520	life	210	quality photo
2504	movie	202	shoot camera
2430	shutter	201	mp camera
2391	month	199	scene mode
2372	button	196	night shot
2363	pics	195	menu system
2348	light	186	battery pack
2299	use	175	battery power
2228	video	172	web site
2212	control	155	camera case
2201	way	151	camera bag
2037	point	145	shirt pocket
2030	people	144	compactflash card
2011	focus	141	shutter release
1952	option	139	lighting condition
1894	money	131	tech support
1829	case	126	red-eye reduction
1784	model	125	slide show
1778	bit	124	burst mode
1734	film	124	lag time
1718	something	122	quality camera

Table C.2.: Top 50 nouns and compound nouns when computing relative frequency using Wikipedia.

Score	Noun	Score	Compound Noun
7.63	5mp	6.61	mb card
6.92	nikon	6.54	macro shot
5.81	easyshare	6.43	auto setting
5.81	aperature	6.17	picture mode
5.76	lense	6.00	download picture
5.66	rechargeables	5.36	focus problem
5.36	lcd	5.32	size camera
5.13	finepix	5.13	quality photo
4.68	alot	5.12	macro mode
4.67	a70	5.09	recycle time
4.62	recharger	5.07	movie mode
4.58	fujifilm	5.07	night picture
4.46	camrea	5.01	photo quality
4.45	smartmedia	5.00	mb memory
4.39	digi-cam	4.95	movie function
4.35	pics	4.95	zoom button
4.29	sony	4.86	focus lock
4.29	digicam	4.66	red-eye reduction
4.25	infolithium	4.64	camera buff
4.25	dissapointment	4.64	battery meter
4.17	sandisk	4.63	medium card
4.17	camaras	4.58	macro picture
4.11	digi	4.58	camera buyer
4.09	fotos	4.57	quality picture
4.09	battries	4.55	shutter delay
4.04	kodak	4.52	auto feature
4.00	soooo	4.46	family camera
4.00	s400	4.46	sd memory
3.99	elph	4.39	idiot proof
3.86	accesories	4.39	picture clarity
3.81	picures	4.39	night portrait
3.81	s30	4.39	kodak camera
3.70	alkalines	4.32	photo mode
3.70	quailty	4.29	battery recharger

Continued on next page

Table C.2: (Continued)

Score	Noun	Score	Compound Noun
3.70	9mp	4.29	smartmedia card
3.64	a60	4.28	view finder
3.58	mc3	4.25	shoot picture
3.58	stabalization	4.25	focus option
3.54	a95	4.25	battery kit
3.50	camara	4.24	camera bag
3.46	mini-tripod	4.21	printer dock
3.46	stablization	4.18	eye reduction
3.46	128mb	4.17	camera dock
3.46	kodaks	4.09	la fotos
3.46	pre-focus	4.09	card error
3.42	amatuer	4.09	camera expert
3.32	powershots	4.09	photography experience
3.32	mpixel	4.09	share button
3.32	apeture	4.05	night mode
3.32	tranfer	4.05	scene mode

Bibliography

Agarwal, A., Biadsy, F., and Mckeown, K. R. (2009). Contextual phrase-level polarity analysis using lexical affect scoring and syntactic n-grams. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics (EACL '09)*, pages 24–32.

Akasaki, S., Yoshinaga, N., and Toyoda, M. (2019). Early discovery of emerging entities in microblogs. *CoRR*, abs/1907.03513.

Alm, C. O., Roth, D., and Sproat, R. (2005). Emotions from text: Machine learning for text-based emotion prediction. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT '05)*, pages 579–586.

Alonso, O. and Baeza-Yates, R. (2011). Design and implementation of relevance assessments using crowdsourcing. In *Proceedings of the 33rd European Conference on Advances in Information Retrieval (ECIR '11)*, pages 153–164.

Alonso, O. and Mizzaro, S. (2009). Can we get rid of TREC assessors? Using mechanical turk for relevance assessment. In *Proceedings of the SIGIR'09 FIRE Workshop*, pages 15–16.

Aly, M. (2005). Survey on multiclass classification methods. *Neural networks*, pages 1–9.

Amigó, E., Artiles, J., Gonzalo, J., Spina, D., Liu, B., and Corujo, A. (2010). WePS-3 evaluation campaign: Overview of the on-line reputation management task. In *CLEF (Notebook Papers/LABs/Workshops)*, volume 1176 of *CEUR Workshop Proceedings*.

Angelidis, S. and Lapata, M. (2018). Summarizing opinions: Aspect extraction meets sentiment prediction and they are both weakly supervised.

Araki, J., Liu, Z., Hovy, E., and Mitamura, T. (2014). Detecting subevent structure for event coreference resolution. In *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC '14)*, pages 4553–4558.

Arora, S., Joshi, M., and Rosé, C. P. (2009). Identifying types of claims in online customer reviews. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics: Short Papers (NAACL '09)*, pages 37–40.

Artiles, J., Borthwick, A., Gonzalo, J., Sekine, S., and Amigó, E. (2010). WePS-3 evaluation campaign: Overview of the web people search clustering and attribute extraction tasks. In *CLEF 2010 LABs and Workshops, Notebook Papers*, volume 1176 of *CEUR Workshop Proceedings*.

Artiles, J., Gonzalo, J., and Sekine, S. (2007). The SemEval-2007 WePS evaluation: Establishing a benchmark for the web people search task. In *Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval '07)*, pages 64–69.

Artiles, J., Gonzalo, J., and Sekine, S. (2009). WePS 2 evaluation campaign: Overview of the web people search clustering task. In *In WePS 2009*.

Aw, A., Zhang, M., Xiao, J., and Su, J. (2006). A phrase-based statistical model for sms text normalization. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions (COLING-ACL '06)*, pages 33–40.

Bagga, A. and Baldwin, B. (1998a). Algorithms for scoring coreference chains. In *In The First International Conference on Language Resources and Evaluation Workshop on Linguistics Coreference*, pages 563–566.

Bagga, A. and Baldwin, B. (1998b). Entity-based cross-document coreferencing using the vector space model. In *Proceedings of the 17th International Conference on Computational Linguistics (COLING '98)*, pages 79–85.

Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Bamman, D., O'Connor, B., and Smith, N. A. (2013). Learning latent personas of film characters. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL '13)*, pages 352–361.

Bamman, D., Underwood, T., and Smith, N. A. (2014). A bayesian mixed effects model of literary character. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL '14)*, pages 370–379.

Banea, C., Mihalcea, R., Wiebe, J., and Hassan, S. (2008). Multilingual subjectivity analysis using machine translation. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP '08)*, pages 127–135.

Banerjee, S. and Pedersen, T. (2002). An adapted lesk algorithm for word sense disambiguation using wordnet. In *Proceedings of the 3rd International Conference on Intelligent Text Processing and Computational Linguistics (CICLing '02)*, pages 136–145.

Beineke, P., Hastie, T., and Vaithyanathan, S. (2004). The sentimental factor: Improving review classification via human-provided information. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL '04)*, pages 263–270.

Bejan, C. A. and Harabagiu, S. (2010). Unsupervised event coreference resolution with rich linguistic features. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL '10)*, pages 1412–1422.

Benamara, F., Chardon, B., Mathieu, Y. Y., and Popescu, V. (2011). Towards context-based subjectivity analysis. In *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP '11)*, pages 1180–1188.

Berend, G. (2011). Opinion expression mining by exploiting keyphrase extraction. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 1162–1170.

Bhagavatula, C., Noraset, T., and Downey, D. (2015). Tabel: Entity linking in web tables. In *The Semantic Web - ISWC 2015*, pages 425–441.

Bhattacharya, I. and Getoor, L. (2006). A latent dirichlet model for unsupervised entity resolution. In *SIAM Conference on Data Mining (SDM)*, pages 47–58.

Björkelund, A. and Farkas, R. (2012). Data-driven multilingual coreference resolution using resolver stacking. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 49–55.

Blair-Goldensohn, S., Neylon, T., Hannan, K., Reis, G. A., Mcdonald, R., and Reynar, J. (2008). Building a sentiment summarizer for local service reviews. In *WWW Workshop on NLP in the Information Explosion Era*.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Bollegala, D., Matsuo, Y., and Ishizuka, M. (2006). Disambiguating personal names on the web using automatically extracted key phrases. *Frontiers in Artificial Intelligence and Applications*, 141:553–557.

Bontcheva, K., Derczynski, L., and Roberts, I. (2017). Crowdsourcing named entity recognition and entity linking corpora. In Ide, N. and Pustejovsky, J., editors, *Handbook of Linguistic Annotation*, pages 875–892. Springer, Dordrecht, Netherlands.

Boser, B. E., Guyon, I. M., and Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the 5th Annual Workshop on Computational Learning Theory (COLT '92)*, pages 144–152.

Branavan, S. R. K., Chen, H., Eisenstein, J., and Barzilay, R. (2009). Learning document-level semantic properties from free-text annotations. *Journal of Artificial Intelligence Research*, 34:569–603.

Brandow, R., Mitze, K., and Rau, L. F. (1995). Automatic condensation of electronic publications by sentence selection. *Information Processing & Management*, 31(5):675–685.

Brenning, A. (2012). Spatial cross-validation and bootstrap for the assessment of prediction rules in remote sensing: the R package 'sperrorest'. In *IEEE International Symposium on Geoscience and Remote Sensing IGARSS*.

Bunescu, R. and Paşca, M. (2006). Using encyclopedic knowledge for named entity disambiguation. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL '06)*, pages 9–16.

Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., and Hullender, G. (2005). Learning to rank using gradient descent. In *Proceedings of the 22Nd International Conference on Machine Learning (ICML '05)*, pages 89–96.

Cai, J. and Strube, M. (2010). End-to-end coreference resolution via hypergraph partitioning. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING '10)*, pages 143–151.

Cao, Z., Qin, T., Liu, T.-Y., Tsai, M.-F., and Li, H. (2007). Learning to rank: From pairwise approach to listwise approach. In *Proceedings of the 24th International Conference on*

Machine Learning (ICML '07), pages 129–136.

Carterette, B., Bennett, P. N., Chickering, D. M., and Dumais, S. T. (2008). Here or there: Preference judgments for relevance. In *Proceedings of the IR Research, 30th European Conference on Advances in Information Retrieval (ECIR '08)*, pages 16–27.

Chaffar, S. and Inkpen, D. (2011). Using a heterogeneous dataset for emotion analysis in text. In *Proceedings of the 24th Canadian Conference on Advances in Artificial Intelligence*, pages 62–67.

Charniak, E. (1972). Toward a model of children's story comprehension. AI TR-266, Massachusetts Institute of Technology Artificial Intelligence Laboratory.

Chen, Y., Hou, W., and Cheng, X. (2018). Hierarchical convolution neural network for emotion cause detection on microblogs. In *Proceedings of the 27th International Conference on Artificial Neural Networks (ICANN '18)*, pages 115–122.

Chen, Y., Lee, S. Y. M., Li, S., and Huang, C.-R. (2010). Emotion cause detection with linguistic constructions. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING '10)*, pages 179–187.

Chen, Z. and Ji, H. (2009). Graph-based event coreference resolution. In *Proceedings of the 2009 Workshop on Graph-based Methods for Natural Language Processing*, pages 54–57.

Chen, Z., Ji, H., and Haralick, R. (2009). A pairwise event coreference model, feature impact and evaluation for event coreference resolution. In *Proceedings of the Workshop on Events in Emerging Text Types*, pages 17–22.

Cheng, X., Chen, Y., Cheng, B., Li, S., and Zhou, G. (2017). An emotion cause corpus for chinese microblogs with multiple-user structures. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 17(1):1–19.

Cheng, X. and Roth, D. (2013). Relational inference for wikification. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP '13)*, pages 1787–1796.

Chinchor, N. A. (1998). Overview of MUC-7/MET-2. In *Proceedings of the 7th Message Understanding Conference (MUC7 '98)*.

Clark, A. and Issco, A. C. (2003). Pre-processing very noisy text. In *Proceedings of the Workshop on Shallow Processing of Large Corpora*.

Clark, E. and Araki, K. (2011). Text normalization in social media: Progress, problems and applications for a pre-processing system of casual english. *Procedia - Social and Behavioral Sciences*, 27:2–11.

Clark, K. and Manning, C. D. (2015). Entity-centric coreference resolution with model stacking. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing: Long Papers (ACL-IJCNLP '15)*, pages 1405–1415.

Clark, K. and Manning, C. D. (2016). Improving coreference resolution by learning entity-level distributed representations. In *Proceedings of the 54th Annual Meeting of the Association* *for Computational Linguistics (ACL '16)*, pages 643–653.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297.

Crammer, K. and Singer, Y. (2002). On the algorithmic implementation of multiclass kernel-based vector machines. *The Journal of Machine Learning Research*, 2:265–292.

Cucerzan, S. (2007). Large-scale named entity disambiguation based on Wikipedia data. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL '07)*, pages 708–716.

Demartini, G., Difallah, D. E., and Cudré-Mauroux, P. (2012). Zencrowd: Leveraging probabilistic reasoning and crowdsourcing techniques for large-scale entity linking. In *Proceedings of the 21st International Conference on World Wide Web (WWW '12)*, pages 469–478.

Denecke, K. (2008). Using sentiwordnet for multilingual sentiment analysis. In *2008 IEEE 24th International Conference on Data Engineering Workshop*, pages 507–512.

Denis, P. and Baldridge, J. (2007). Joint determination of anaphoricity and coreference resolution using integer programming. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL '07)*, pages 236–243.

Denis, P. and Baldridge, J. (2008). Specialized models and ranking for coreference resolution. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP '08)*, pages 660–669.

Dietterich, T. G. (1998). Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, 10(7):1895–1923.

Difallah, D. E., Demartini, G., and Cudré-Mauroux, P. (2012). Mechanical cheat: Spamming schemes and adversarial techniques on crowdsourcing platforms. In *Proceedings of the 1st International Workshop on Crowdsourcing Web Search*, volume 842 of *CEUR Workshop Proceedings*, pages 26–30.

Ding, X., Liu, B., and Yu, P. S. (2008). A holistic lexicon-based approach to opinion mining. In *Proceedings of the 2008 International Conference on Web Search and Data Mining (WSDM '08)*, pages 231–240.

Doddington, G., Mitchell, A., Przybocki, M., Ramshaw, L., Strassel, S., and Weischedel, R. (2004). The automatic content extraction (ACE) program – tasks, data, and evaluation. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC '04)*.

Doran, W., Stokes, N., Carthy, J., and Dunnion, J. (2004). Comparing lexical chain-based summarisation approaches using an extrinsic evaluation. In *Proceedings of the 5th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing '04)*, pages 112–117.

dos Santos, C. N. and Gatti, M. (2014). Deep convolutional neural networks for sentiment analysis of short texts. In *Proceedings of the 25th International Conference on Computational Linguistics (COLING '14)*, pages 69–78.

Downs, J. S., Holbrook, M. B., Sheng, S., and Cranor, L. F. (2010). Are your participants gaming the system?: Screening mechanical turk workers. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10)*, pages 2399–2402.

Dragoni, M., Da Costa Pereira, C., Tettamanzi, A. G. B., and Villata, S. (2018). Combining Argumentation and Aspect-Based Opinion Mining: The SMACk System. *AI Communications*, 31(1):75 – 95.

Dredze, M., McNamee, P., Rao, D., Gerber, A., and Finin, T. (2010). Entity disambiguation for knowledge base population. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING '10)*, pages 277–285.

Dutta, S. and Weikum, G. (2015). Cross-document co-reference resolution using sample-based clustering with knowledge enrichment. *Transactions of the Association for Computational Linguistics (TACL '15)*, 3:15–28.

Eickhoff, C. and Vries, A. P. (2013). Increasing cheat robustness of crowdsourcing tasks. *Information Retrieval*, 16(2):121–137.

Ekman, P. (1972). Universals and cultural differences in facial expressions of emotion. *Nebraska Symposium on Motivation*, 19:207–282.

Eriksson, K. and Simpson, B. (2010). Emotional reactions to losing explain gender differences in entering a risky lottery. *Judgment and Decision Making*, 5(3):159–163.

Eshel, Y., Cohen, N., Radinsky, K., Markovitch, S., Yamada, I., and Levy, O. (2017). Named entity disambiguation for noisy text. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 58–68.

Esquivel, J., Albakour, M.-D., Martínez, M., Corney, D., and Moussa, S. (2017). On the long-tail entities in news. In *Bibliometric-Enhanced Information Retrieval: 5th International BIR Workshop*, pages 691–697.

Even-Zohar, Y. and Roth, D. (2001). A sequential model for multi-class classification. In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing (EMNLP '01)*.

Färber, M., Rettinger, A., and El Asmar, B. (2016). On emerging entity detection. In Blomqvist, E., Ciancarini, P., Poggi, F., and Vitali, F., editors, *Knowledge Engineering and Knowledge Management*, pages 223–238.

Faulkner, A. R. (2014). *Automated Classification of Argument Stance in Student Essays: A Linguistically Motivated Approach with an Application for Supporting Argument Summarization.* PhD thesis, The City University of New York. CUNY Academic Works: `https://academicworks.cuny.edu/gc_etds/204`.

Ferraro, F., Thomas, M., Gormley, M. R., Wolfe, T., Harman, C., and Van Durme, B. (2014). Concretely annotated corpora. In *The NIPS 2014 AKBC Workshop*.

Fetahu, B., Anand, A., and Anand, A. (2015). How much is wikipedia lagging behind news? *Proceedings of the ACM Web Science Conference on ZZZ - WebSci '15*.

Finkel, J. R., Grenager, T., and Manning, C. (2005). Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL '05)*, pages 363–370.

Fleischman, M. and Hovy, E. (2002). Fine grained classification of named entities. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING '02)*, pages 1–7.

Fleischman, M. and Hovy, E. (2004). Multi-document person name resolution. In *Workshop on Reference Resolution and its Applications.*

Fleiss, J. L. (1971). Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382.

Forman, G. and Scholz, M. (2010). Apples-to-apples in cross-validation studies: Pitfalls in classifier performance measurement. *SIGKDD Explorations Newsletter*, 12(1):49–57.

Francis-Landau, M., Durrett, G., and Klein, D. (2016). Capturing semantic similarity for entity linking with convolutional neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT '16)*, pages 1256–1261.

Freund, Y., Iyer, R., Schapire, R. E., and Singer, Y. (2003). An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, 4:933–969.

Fukushima, K. (1980). Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36:193–202.

Gale, W. A., Church, K. W., and Yarowsky, D. (1992). Work on statistical methods for word sense disambiguation. In *Working Notes of the AAAI Fall Symposium on Probabilistic Approaches to Natural Language*, pages 54–60.

Gamon, M., Aue, A., Corston-Oliver, S., and Ringger, E. (2005). Pulse: Mining customer opinions from free text. In *Advances in Intelligent Data Analysis VI*, pages 121–132.

Garigliotti, D., Albakour, D., Martinez, M., and Balog, K. (2019). Unsupervised context retrieval for long-tail entities. In *Proceedings of the 2019 ACM SIGIR International Conference on Theory of Information Retrieval (ICTIR '19)*, pages 225–228.

Ghazi, D., Inkpen, D., and Szpakowicz, S. (2015). Detecting emotion stimuli in emotion-bearing sentences. In *Computational Linguistics and Intelligent Text Processing*, pages 152–165.

Gildea, D. and Jurafsky, D. (2002). Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Gillick, D. (2009). Sentence boundary detection and the problem with the u.s. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics: Short Papers (NAACL '09)*, pages 241–244.

Glaser, A. and Kuhn, J. (2014). Exploring the utility of coreference chains for improved identification of personal names. In *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC '14)*, pages 2570–2577.

Glaser, A. and Kuhn, J. (2016). Named entity disambiguation for little known referents: a topic-based approach. In *Proceedings of the 26th International Conference on Computational Linguistics (COLING '16)*, pages 1481–1492.

Glaser, A. and Schütze, H. (2012). Automatic generation of short informative sentiment summaries. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL '12)*, pages 276–285.

Globerson, A., Lazic, N., Chakrabarti, S., Subramanya, A., Ringgaard, M., and Pereira, F. (2016). Collective entity resolution with multi-focal attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL '16)*, pages 621–631.

Gooi, C. H. and Allan, J. (2004). Cross-document coreference on a large scale corpus. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL '04)*, pages 9–16.

Gottipati, S. and Jiang, J. (2011). Linking entities to a knowledge base with query expansion. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP '11)*, pages 804–813.

Graff, D. and Cieri, C. (2003). English Gigaword LDC2003T05. Linguistic Data Consortium, Philadelphia.

Graves, A., Mohamed, A., and Hinton, G. E. (2013). Speech recognition with deep recurrent neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013*, pages 6645–6649.

Grishman, R. and Sundheim, B. (1995). Design of the MUC-6 evaluation. In *Proceedings of the 6th Conference on Message Understanding (MUC6 '95)*, pages 1–11.

Grishman, R. and Sundheim, B. (1996). Message understanding conference-6: A brief history. In *Proceedings of the 16th Conference on Computational Linguistics (COLING '96)*, pages 466–471.

Gruetze, T., Kasneci, G., Zuo, Z., and Naumann, F. (2016). Coheel: Coherent and efficient named entity linking through random walks. *Web Semantics: Science, Services and Agents on the World Wide Web*, 37(0).

Gui, L., Wu, D., Xu, R., Lu, Q., and Zhou, Y. (2016). Event-driven emotion cause extraction with corpus construction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1639–1649.

Gui, L., Yuan, L., Xu, R., Liu, B., Lu, Q., and Zhou, Y. (2014). Emotion cause detection with linguistic construction in chinese weibo text. *Communications in Computer and Information Science*, 496:457–464.

Guo, J., Xu, G., Cheng, X., and Li, H. (2009). Named entity recognition in query. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '09)*, page 267–274.

Guo, S., Chang, M.-W., and Kiciman, E. (2013). To link or not to link? a study on end-to-end tweet entity linking. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*

(NAACL-HLT '15), pages 1020–1030.

Guo, Z. and Barbosa, D. (2014a). Entity linking with a unified semantic representation. In *Proceedings of the 23rd International Conference on World Wide Web (WWW '14)*, pages 1305–1310.

Guo, Z. and Barbosa, D. (2014b). Robust entity linking via random walks. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management (CIKM '14)*, pages 499–508.

Hachey, B., Radford, W., and Curran, J. R. (2011). Graph-based named entity linking with wikipedia. In *Proceedings of the 12th International Conference on Web Information System Engineering (WISE '11)*, pages 213–226.

Hachey, B., Radford, W., Nothman, J., Honnibal, M., and Curran, J. R. (2013). Evaluating entity linking with wikipedia. *Artificial Intelligence*, 194:130–150.

Haghighi, A. and Klein, D. (2007). Unsupervised coreference resolution in a nonparametric bayesian model. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL '07)*, pages 848–855.

Hahm, Y., Park, J., Lim, K., Kim, Y., Hwang, D., and Choi, K.-S. (2014). Named entity corpus construction using wikipedia and dbpedia ontology. In *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC '14)*, pages 2565–2569.

Hahn, U., Tomanek, K., Beisswanger, E., and Faessler, E. (2010). A proposal for a configurable silver standard. In *Proceedings of the 4th Linguistic Annotation Workshop (LAW IV '10)*, pages 235–242.

Hajishirzi, H., Zilles, L., Weld, D. S., and Zettlemoyer, L. S. (2013). Joint coreference resolution and named-entity linking with multi-pass sieves. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP '13)*, pages 289–299.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The weka data mining software: An update. *SIGKDD Explorations*, 11(1):10–18.

Han, H., Giles, L., Zha, H., Li, C., and Tsioutsiouliklis, K. (2004). Two supervised learning approaches for name disambiguation in author citations. In *Proceedings of the 4th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL '04)*, pages 296–305.

Han, X. and Sun, L. (2012). An entity-topic model for entity linking. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL '12)*, pages 105–115.

Han, X., Sun, L., and Zhao, J. (2011). Collective entity linking in web text: A graph-based method. In *Proceedings of the 34th Annual International ACM SIGIR Conference on Research and Development on Information Retrieval (SIGIR '11)*, pages 765–774.

Han, X. and Zhao, J. (2009). Named entity disambiguation by leveraging wikipedia semantic knowledge. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM '09)*, pages 215–224.

Hand, D. J. and Yu, K. (2001). Idiot's bayes—not so stupid after all? *International Statistical*

Review, 69(3):385–398.

Hastie, T. and Tibshirani, R. (1998). Classification by pairwise coupling. In *Proceedings of the 1997 Conference on Advances in Neural Information Processing Systems 10 (NIPS '97)*, pages 507–513.

Hatzivassiloglou, V. and McKeown, K. R. (1997). Predicting the semantic orientation of adjectives. In *Proceedings of the 8th Conference of the European Chapter of the Association for Computational Linguistics (EACL '97)*, pages 174–181.

He, Z., Liu, S., Li, M., Zhou, M., Zhang, L., and Wang, H. (2013). Learning entity representation for entity disambiguation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: Short Papers (ACL '13)*, pages 30–34.

Hirao, T., Sasaki, Y., and Isozaki, H. (2001). Advantages of query biased summaries in information retrieval. In *Prodeedings of NAACL 2001 workshop on Automatic Summarization.*

Hirschman, L. and Chinchor, N. (1998). Appendix F: MUC-7 coreference task definition (version 3.0). In *Proceedings of the 7th Message Understanding Conference (MUC7 '98).*

Hobbs, J. R. (1976). Pronoun resolution. Research report 76-1, Department of Computer Sciences, City College, City University of New York.

Hobbs, J. R. (1978). Resolving pronoun references. *Lingua*, 44:311–338.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Hoffart, J., Altun, Y., and Weikum, G. (2014). Discovering emerging entities with ambiguous names. In *Proceedings of the 23rd International Conference on World Wide Web (WWW '14)*, pages 385–396.

Hoffart, J., Milchevski, D., Weikum, G., Anand, A., and Singh, J. (2016). The knowledge awakens: Keeping knowledge bases fresh with emerging entities. In *Proceedings of the 25th International Conference Companion on World Wide Web (WWW '16 Companion)*, pages 203–206.

Hoffart, J., Seufert, S., Nguyen, D. B., Theobald, M., and Weikum, G. (2012). Kore: Keyphrase overlap relatedness for entity disambiguation. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management (CIKM '12)*, pages 545–554.

Hoffart, J., Yosef, M. A., Bordino, I., Fürstenau, H., Pinkal, M., Spaniol, M., Taneva, B., Thater, S., and Weikum, G. (2011). Robust disambiguation of named entities in text. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP '11)*, pages 782–792.

Hofmann, T. (1999). Probabilistic latent semantic analysis. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence (UAI'99)*, pages 289–296.

Horton, J. J., Rand, D. G., and Zeckhauser, R. J. (2011). The online laboratory: conducting experiments in a real labor market. *Experimental Economics*, 14(3):399–425.

Hou, Y., Markert, K., and Strube, M. (2013). Global inference for bridging anaphora resolution. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for*

Computational Linguistics: Human Language Technologies (NAACL-HLT '13), pages 907–917.

Hou, Y., Markert, K., and Strube, M. (2014). A rule-based system for unrestricted bridging resolution: Recognizing bridging anaphora and finding links to antecedents. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP '14)*, pages 2082–2093.

Hovy, E., Lin, C., and Zhou, L. (2005). Evaluating DUC 2005 using basic elements. In *Proceedings of the 5th Document Understanding Conference (DUC)*.

Hovy, E., Lin, C., Zhou, L., and Fukumoto, J. (2006a). Automated summarization evaluation with basic elements. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC '06)*.

Hovy, E., Marcus, M., Palmer, M., Ramshaw, L., and Weischedel, R. (2006b). OntoNotes: The 90% solution. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: Short Papers (NAACL '06)*, pages 57–60.

Hsu, C.-W. and Lin, C.-J. (2002). A comparison of methods for multiclass support vector machines. *Transactions on Neural Networks*, 13(2):415–425.

Hu, M. and Liu, B. (2004a). Mining opinion features in customer reviews. In *Proceedings of the 19th National Conference on Artifical Intelligence (AAAI'04)*, pages 755–760.

Hu, M. and Liu, B. (2004b). Mining and summarizing customer reviews. In *Proceedings of the 10th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '04)*, pages 168–177.

Hu, M. and Liu, B. (2004c). Mining and summarizing customer reviews. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '04)*, pages 168–177.

Huang, S., Niu, Z., and Shi, C. (2014). Automatic construction of domain-specific sentiment lexicon based on constrained label propagation. *Knowledge-Based Systems*, 56:191 – 200.

Huddleston, R. D. and Pullum, G. K. (2002). *The Cambridge Grammar of the English Language*. Cambridge textbooks in linguistics. Cambridge University Press.

Ikeda, M., Yoshida, M., Ono, S., Nakagawa, H., and Sato, I. (2009). Person name disambiguation on the web by two-stage clustering. In *WePS 2009, 18th WWW Conference*.

Ilievski, F., Postma, M., and Vossen, P. (2016). Semantic overfitting: what 'world' do we consider when evaluating disambiguation of text? In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1180–1191.

Ilievski, F., Vossen, P., and Schlobach, S. (2018). Systematic study of long tail phenomena in entity linking. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING '18)*, pages 664–674.

Ilievski, F., Vossen, P., and van Erp, M. (2017). Hunger for contextual knowledge and a road map to intelligent entity linking. In *Language, Data, and Knowledge*, pages 143–149.

Ipeirotis, P. G. (2010a). Demographics of mechanial turk. CeDER-10–01 working paper. New York University.

Ipeirotis, P. G. (2010b). Be a top mechanical turk worker: You need \$5 and 5 minutes. `http://www.behind-the-enemy-lines.com/2010/10/be-top-mechanical-turk-worker-you-need.html`. Accessed: July 26, 2017.

Ipeirotis, P. G., Provost, F., and Wang, J. (2010). Quality management on amazon mechanical turk. In *Proceedings of the ACM SIGKDD Workshop on Human Computation (HCOMP '10)*, pages 64–67.

Isozaki, H. and Kazawa, H. (2002). Efficient support vector classifiers for named entity recognition. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING '02)*, pages 1–7.

Jakob, N. and Gurevych, I. (2010). Extracting opinion targets in a single- and cross-domain setting with conditional random fields. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP '10)*, pages 1035–1045.

Ji, H., Grishman, R., Dang, H. T., Griffitt, K., and Ellis, J. (2010). Overview of the TAC 2010 knowledge base population track. In *Proceedings of the 3rd Text Analysis Conference (TAC '10)*.

Jia, L., Yu, C., and Meng, W. (2009). The effect of negation on sentiment analysis and retrieval effectiveness. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM '09)*, pages 1827–1830.

Jiang, L., Yu, M., Zhou, M., Liu, X., and Zhao, T. (2011). Target-dependent twitter sentiment classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT '11)*, pages 151–160.

Jijkoun, V., Khalid, M. A., Marx, M., and de Rijke, M. (2008). Named entity normalization in user generated content. In *Proceedings of the 2nd Workshop on Analytics for Noisy Unstructured Text Data (AND '08)*, pages 23–30.

Jin, J., Ji, P., and Kwong, C. (2016). What makes consumers unsatisfied with your products. *Eng. Appl. Artif. Intell.*, 47(C):38–48.

Jindal, N. and Liu, B. (2008). Opinion spam and analysis. In *Proceedings of the International Conference on Web search and Web Wata Mining (WSDM '08)*, pages 219–230.

Jing, H., Barzilay, R., McKeown, K., and Elhadad, M. (1998). Summarization evaluation methods: Experiments and analysis. In *AAAI Symposium on Intelligent Summarization*, pages 60–68.

Joachims, T. (2002). Evaluating retrieval performance using clickthrough data. *SIGIR Workshop on Mathematical/Formal Methods in Information Retrieval.*

John, G. H. and Langley, P. (1995). Estimating continuous distributions in bayesian classifiers. In *11th Conference on Uncertainty in Artificial Intelligence*, pages 338–345.

Jones, K. S. and Galliers, J. R. (1996). *Evaluating Natural Language Processing Systems: An Analysis and Review.* Springer-Verlag, Secaucus, NJ, USA.

Julio, S. and Ayu, P. (2017). Emotion classification on youtube comments using word embedding. In *2017 International Conference on Advanced Informatics, Concepts, Theory, and Applications (ICAICTA)*, pages 1–5.

Jurafsky, D. and Martin, J. H. (2009). *Speech and Language Processing.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2nd edition.

Kang, N., van Mulligen, E. M., and Kors, J. A. (2012). Training text chunkers on a silver standard corpus: can silver replace gold? *BMC Bioinformatics*, 13:17.

Karimi, S., Yin, J., and Baum, J. (2015). Evaluation methods for statistically dependent text. *Computational Linguistics*, 41(3):539–548.

Kataria, S. S., Kumar, K. S., Rastogi, R. R., Sen, P., and Sengamedu, S. H. (2011). Entity disambiguation with hierarchical topic models. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '11)*, pages 1037–1045.

Kaufmann, N., Schulze, T., and Veit, D. J. (2011). More than fun and money. worker motivation in crowdsourcing - a study on mechanical turk. In *Proceedings of the 17th Americas Conference on Information Systems (AMCIS 2011)*, page Paper 340.

Keegan, B., Gergle, D., and Contractor, N. (2013). Hot off the wiki: Structures and dynamics of wikipedia's coverage of breaking news events. *American Behavioral Scientist*, 57(5):595–622.

Keerthi, S., Shevade, S., Bhattacharyya, C., and Murthy, K. (2001). Improvements to platt's smo algorithm for svm classifier design. *Neural Computation*, 13(3):637–649.

Khalid, M. A., Jijkoun, V., and de Rijke, M. (2008). The impact of named entity normalization on information retrieval for question answering. In *Advances in Information Retrieval*, pages 705–710.

Kim, S., Zhang, J., Chen, Z., Oh, A., and Liu, S. (2013). A hierarchical aspect-sentiment model for online reviews. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence (AAAI '13)*, pages 526–533.

Kim, S.-M. and Hovy, E. (2004). Determining the sentiment of opinions. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING '04)*.

Kim, S.-M. and Hovy, E. (2006). Automatic identification of pro and con reasons in online reviews. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions (COLING-ACL '06)*, pages 483–490.

Kim, Y. (2014). Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP '14)*, pages 1746–1751.

Kittur, A., Chi, E. H., and Suh, B. (2008). Crowdsourcing user studies with mechanical turk. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)*, pages 453–456.

Klapholz, D. and Lockman, A. (1975). Contextual reference resolution. American Journal of Computational Linguistics, microfiche 36.

Kobayashi, N., Inui, K., Matsumoto, Y., Tateishi, K., and Fukushima, T. (2004). Collecting evaluative expressions for opinion extraction. In *Proceedings of the International Joint Conference on Natural Language Processing (IJCNLP '04)*, pages 596–605.

Kravalová, J. and Žabokrtský, Z. (2009). Czech named entity corpus and svm-based recognizer. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration (NEWS '09)*, pages 194–201.

Ku, L. W., Liang, Y. T., and Chen, H. H. (2006). Opinion extraction, summarization and tracking in news and blog corpora. In *Proceedings of AAAI-2006 Spring Symposium on Computational Approaches to Analyzing Weblogs.*

Kulkarni, A. and Pedersen, T. (2005). Name discrimination and email clustering using unsupervised clustering and labeling of similar contexts. In *Proceedings of the 2nd Indian International Conference on Artificial Intelligence*, pages 703–722.

Kulkarni, A. K. (2005). Unsupervised discrimination and labeling of ambiguous names. In *Proceedings of the ACL Student Research Workshop*, pages 145–150.

Kulkarni, A. K. (2006). Unsupervised context discrimination and automatic cluster stopping. Master of science thesis, Department of Computer Science, University of Minnesota, Duluth.

Kulkarni, S., Singh, A., Ramakrishnan, G., and Chakrabarti, S. (2009). Collective annotation of wikipedia entities in web text. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '09)*, pages 457–466.

Kupiec, J., Pedersen, J., and Chen, F. (1995). A trainable document summarizer. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development on Information Retrieval (SIGIR '95)*, pages 68–73.

Lakoff, G. (1968). Pronouns and reference. Distributed by Indiana University Linguistics Club, Bloomington.

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., and Dyer, C. (2016). Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT '16)*, pages 260–270.

Landauer, T. K. and Dutnais, S. T. (1997). A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, 104(2):211–240.

Lawrence, J. and Reed, C. (2019). Argument mining: A survey. *Computational Linguistics*, pages 1–54.

Laws, F., Scheible, C., and Schütze, H. (2011). Active learning with amazon mechanical turk. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP '11)*, pages 1546–1556.

Lazic, N., Subramanya, A., Ringgaard, M., and Pereira, F. (2015). Plato: A selective context model for entity resolution. *Transactions of the Association for Computational Linguistics (TACL '15)*, 3:503–515.

Le, J., Edmonds, A., Hester, V., and Biewald, L. (2010). Ensuring quality in crowdsourced search relevance evaluation: The effects of training question distribution. In *In SIGIR 2010 Workshop*, pages 21–26.

Lee, H., Chang, A., Peirsman, Y., Chambers, N., Surdeanu, M., and Jurafsky, D. (2013). Deterministic coreference resolution based on entity-centric, precision-ranked rules. *Computational Linguistics*, 39(4):885–916.

Lee, H., Peirsman, Y., Chang, A., Chambers, N., Surdeanu, M., and Jurafsky, D. (2011). Stanford's multi-pass sieve coreference resolution system at the conll-2011 shared task. In *Proceedings of the 15th Conference on Computational Natural Language Learning: Shared Task*, pages 28–34.

Lee, K., He, L., Lewis, M., and Zettlemoyer, L. (2017). End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP '17)*, pages 188–197.

Lee, S. Y. M., Chen, Y., and Huang, C.-R. (2010). A text-driven rule-based system for emotion cause detection. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, pages 45–53.

Levi, J. (1978). *The syntax and semantics of complex nominals.* Academic Press.

Li, S., Huang, L., Wang, R., and Zhou, G. (2015). Sentence-level emotion classification with label and context dependence. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1045–1053.

Li, X., Feng, S., Wang, D., and Zhang, Y. (2019). Context-aware emotion cause analysis with multi-attention-based neural network. *Knowledge-Based Systems*, 174:205–218.

Li, Y., Wang, C., Han, F., Han, J., Roth, D., and Yan, X. (2013). Mining evidences for named entity disambiguation. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '13)*, pages 1070–1078.

Limaye, G., Sarawagi, S., and Chakrabarti, S. (2010). Annotating and searching web tables using entities, types and relationships. *Proceedings of VLDB Endowment*, 3(1-2):1338–1347.

Lin, C.-Y. (2004). ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81.

Lin, C.-Y. and Hovy, E. (2003). Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (NAACL '03)*, pages 71–78.

Lin, T., Mausam, and Etzioni, O. (2012). Entity linking at web scale. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX '12)*, pages 84–88.

Ling, X. and Weld, D. S. (2012). Fine-grained entity recognition. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence (AAAI '12)*, pages 94–100.

Lippi, M. and Torroni, P. (2016). Argumentation mining: State of the art and emerging trends. *ACM Transactions on Internet Technology*, 16(2):10:1–10:25.

Liu, B. (2010). Sentiment analysis and subjectivity. In Indurkhya, N. and Damerau, F. J., editors, *Handbook of Natural Language Processing*, pages 627–666. Chapman & Hall/CRC, 2nd edition.

Liu, B. (2011). *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data*. Springer-Verlag, Berlin Heidelberg, 2nd edition.

Liu, B. (2015). *Sentiment Analysis: Mining Opinions, Sentiments, and Emotions*. Cambridge University Press.

Liu, X., Zhou, M., Wei, F., Fu, Z., and Zhou, X. (2012). Joint inference of named entity recognition and normalization for tweets. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL '12)*, pages 526–535.

Lloret, E., Balahur, A., Gómez, J., Montoyo, A., and Sanz, M. (2012). Towards a unified framework for opinion retrieval, mining and summarization. *Journal of Intelligent Information Systems*, 39(3):711–747.

Lu, Y., Castellanos, M., Dayal, U., and Zhai, C. (2011). Automatic construction of a context-aware sentiment lexicon: An optimization approach. In *Proceedings of the 20th International Conference on World Wide Web (WWW '11)*, pages 347–356.

Lubani, M. and Mohd Noah, S. (2018). Building compact entity embeddings using wikidata. *International Journal on Advanced Science, Engineering and Information Technology*, 8(4-2):1437–1445.

Luo, X. (2005). On coreference resolution performance metrics. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT '05)*, pages 25–32.

Luo, X., Ittycheriah, A., Jing, H., Kambhatla, N., and Roukos, S. (2004). A mention-synchronous coreference resolution algorithm based on the bell tree. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL '04)*, pages 135–142.

López, R. and Pardo, T. A. S. (2015). Experiments on sentence boundary detection in user-generated web content. In *Proceedings of the 16th International Conference on Intelligent Text Processing and Computational Linguistics*, pages 227–237.

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., and Potts, C. (2011). Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT '11)*, pages 142–150.

Mani, I. and Bloedorn, E. (1997). Multi-document summarization by graph search and matching. In *Proceedings of the 14th National Conference on Artificial Intelligence and 9th Innovative Applications of Artificial Intelligence Conference, (AAAI-IAAI '97)*, pages 622–628.

Mann, G. S. and Yarowsky, D. (2003). Unsupervised personal name disambiguation. In *Pro-*

ceedings of the 7th Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4 (CoNLL '03), pages 33–40.

Manning, C. and Klein, D. (2003). Optimization, maxent models, and conditional estimation without magic. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology: Tutorials (NAACL '03)*, pages 8–8.

Manning, C. D., Raghavan, P., and Schütze, H. (2008). *Introduction to Information Retrieval.* Cambridge University Press, New York, NY, USA.

Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S. J., and McClosky, D. (2014). The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations (ACL '14)*, pages 55–60.

Marge, M., Banerjee, S., and Rudnicky, A. I. (2010). Using the amazon mechanical turk to transcribe and annotate meeting speech for extractive summarization. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk (CSLDAMT '10)*, pages 99–107.

Markert, K., Hou, Y., and Strube, M. (2012). Collective classification for fine-grained information status. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL '12)*, pages 795–804.

Mason, W. and Watts, D. J. (2010). Financial incentives and the "performance of crowds". *SIGKDD Explorations Newsletter*, 11(2):100–108.

McCallum, A. and Li, W. (2003). Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the 7th Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4 (CoNLL '03)*, pages 188–191.

McCallum, A. and Wellner, B. (2003). Toward conditional models of identity uncertainty with application to proper noun coreference. In *Proceedings of the 2003 International Conference on Information Integration on the Web (IIWEB '03)*, pages 79–84.

McCallum, A. and Wellner, B. (2004). Conditional models of identity uncertainty with application to noun coreference. In *Proceedings of the 17th International Conference on Neural Information Processing Systems (NIPS '04)*, pages 905–912.

McCallum, A. K. (2002). MALLET: A machine learning for language toolkit. `http://mallet.cs.umass.edu/`.

McCarthy, J. F. and Lehnert, W. G. (1995). Using decision trees for coreference resolution. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pages 1050–1055.

McCreadie, R. M. C., Macdonald, C., and Ounis, I. (2010). Crowdsourcing a news query classification dataset. In *In Proceedings of the ACM SIGIR 2010 Workshop on Crowdsourcing for Search Evaluation (CSE 2010)*, pages 31–38.

McNamee, P., Simpson, H., and Dang, H. T. (2009). Overview of the TAC 2009 knowledge base population track. In *Proceedings of the 2009 Text Analysis Conference*.

Meena, A. and Prabhakar, T. V. (2007). Sentence level sentiment analysis in the presence of conjuncts using linguistic analysis. In *Proceedings of the 29th European Conference on IR Research (ECIR '07)*, pages 573–580.

Melville, P., Gryc, W., and Lawrence, R. D. (2009). Sentiment analysis of blogs by combining lexical knowledge with text classification. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '09)*, pages 1275–1284.

Meng, X., Wei, F., Liu, X., Zhou, M., Li, S., and Wang, H. (2012). Entity-centric topic-oriented opinion summarization in twitter. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '12)*, pages 379–387.

Michie, D., Spiegelhalter, D. J., Taylor, C. C., and Campbell, J., editors (1994). *Machine Learning, Neural and Statistical Classification.* Ellis Horwood, Upper Saddle River, NJ, USA.

Mihalcea, R. and Csomai, A. (2007). Wikify!: Linking documents to encyclopedic knowledge. In *Proceedings of the 16th ACM Conference on Conference on Information and Knowledge Management (CIKM '07)*, pages 233–242.

Miller, G. A. (1995). Wordnet: A lexical database for english. *Communications of the ACM*, 38(11):39–41.

Milne, D. and Witten, I. H. (2008). Learning to link with wikipedia. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM '08)*, pages 509–518.

Minel, Jean-Luc, N. S. and Piat, G. (1997). How to appreciate the quality of automatic text summarization? examples of FAN and MLUCE protocols and their results on SERAPHIN. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics (ACL '97)*, pages 25–30.

Mishne, G. (2005). Experiments with mood classification in blog posts. In *1st Workshop on Stylistic Analysis Of Text For Information Access.*

Mitchell, T. M. (1997). *Machine Learning.* McGraw-Hill, Inc., New York, NY, USA, 1st edition.

Moldovan, D. I. and Novischi, A. (2004). Word sense disambiguation of wordnet glosses. *Computer Speech and Language*, 18(3):301–317.

Monahan, S., Lehmann, J., Nyberg, T., Plymale, J., and Jung, A. (2011). Cross-lingual cross-document coreference with entity linking. In *Proceedings of the 4th Text Analysis Conference (TAC '11).*

Moraes, R., Valiati, J. a. F., and GaviãO Neto, W. P. (2013). Document-level sentiment classification: An empirical comparison between svm and ann. *Expert Systems with Applications*, 40(2):621–633.

Morinaga, S., Yamanishi, K., Tateishi, K., and Fukushima, T. (2002). Mining product reputations on the web. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '02)*, pages 341–349.

Moro, A., Raganato, A., and Navigli, R. (2014). Entity linking meets word sense disambiguation: a unified approach. *Transactions of the Association for Computational Linguistics*, 2:231–244.

Mueller, D. and Durrett, G. (2018). Effective use of context in noisy entity linking. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1024–1029.

Murty, M. N. and Devi, V. S. (2011). *Pattern Recognition - An Algorithmic Approach.* Undergraduate Topics in Computer Science. Springer.

Nadeau, D. and Sekine, S. (2007). A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26.

Nakashole, N., Tylenda, T., and Weikum, G. (2013). Fine-grained semantic typing of emerging entities. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL '13)*, pages 1488–1497.

Nakov, P., Ritter, A., Rosenthal, S., Sebastiani, F., and Stoyanov, V. (2016). Semeval-2016 task 4: Sentiment analysis in twitter. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval '16)*, pages 1–18.

Napoles, C., Gormley, M., and Van Durme, B. (2012). Annotated gigaword. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX '12)*, pages 95–100.

Nasukawa, T. and Yi, J. (2003). Sentiment analysis: Capturing favorability using natural language processing. In *Proceedings of the 2nd International Conference on Knowledge Capture (K-CAP '03)*, pages 70–77.

Navigli, R. (2009). Word sense disambiguation: A survey. *ACM Computing Surveys*, 41(2):10:1–10:69.

Nemeskey, D. M. and Simon, E. (2012). Automatically generated ne tagged corpora for english and hungarian. In *Proceedings of the 4th Named Entity Workshop (NEWS '12)*, pages 38–46.

Nenkova, A. and Passonneau, R. J. (2004). Evaluating content selection in summarization: The pyramid method. In *Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL '04)*, pages 145–152.

Nenkova, A., Passonneau, R. J., and McKeown, K. (2007). The pyramid method: Incorporating human content selection variation in summarization evaluation. *Transactions on Speech and Language Processing*, 4(2):4.

Neviarouskaya, A. and Aono, M. (2013). Extracting causes of emotions from text. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 932–936.

Newman, D., Asuncion, A., Smyth, P., and Welling, M. (2009). Distributed algorithms for topic models. *Journal of Machine Learning Research*, 10:1801–1828.

Ng, V. (2010). Supervised noun phrase coreference research: The 1st fifteen years. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL '10)*, pages 1396–1411.

Ng, V. (2017). Machine learning for entity coreference resolution: A retrospective look at two decades of research. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI '17)*, pages 4877–4884.

Ng, V. and Cardie, C. (2002). Improving machine learning approaches to coreference resolution. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL '02)*, pages 104–111.

Nguyen, D. B., Hoffart, J., Theobald, M., and Weikum, G. (2014). Aida-light: High-throughput named-entity disambiguation. In *Proceedings of the Workshop on Linked Data on the Web co-located with the 23rd International World Wide Web Conference*, CEUR Workshop Proceedings.

Nguyen, H. T. and Cao, T. H. (2008). Named entity disambiguation: A hybrid statistical and rule-based incremental approach. In *The Semantic Web: 3rd Asian Semantic Web Conference, ASWC 2008, Bangkok, Thailand, December 8-11, 2008. Proceedings.*, pages 420–433. Springer, Berlin, Heidelberg.

Nothman, J., Curran, J. R., and Murphy, T. (2008). Transforming wikipedia into named entity training data. In *In Proceedings of the Australasian Language Technology Association Workshop 2008*, pages 124–132.

Nothman, J., Ringland, N., Radford, W., Murphy, T., and Curran, J. R. (2013). Learning multilingual named entity recognition from wikipedia. *Artificial Intelligence*, 194(0):151–175.

Nowak, S. and Rüger, S. (2010). How reliable are annotations via crowdsourcing: A study about inter-annotator agreement for multi-label image annotation. In *Proceedings of the International Conference on Multimedia Information Retrieval (MIR '10)*, pages 557–566.

O'Hare, N., Davy, M., Bermingham, A., Ferguson, P., Sheridan, P., Gurrin, C., and Smeaton, A. F. (2009). Topic-dependent sentiment analysis of financial blogs. In *Proceedings of the 1st International CIKM Workshop on Topic-sentiment Analysis for Mass Opinion (TSA '09)*, pages 9–16.

Page, L., Brin, S., Motwani, R., and Winograd, T. (1998). The pagerank citation ranking: Bringing order to the web. In *Proceedings of the 7th International World Wide Web Conference*, pages 161–172.

Palmer, D. D. and Hearst, M. A. (1997). Adaptive multilingual sentence boundary disambiguation. *Computational Linguistics*, 23(2):241–267.

Pang, B. and Lee, L. (2004). A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL '04)*.

Pang, B. and Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135.

Pang, B., Lee, L., and Vaithyanathan, S. (2002). Thumbs up?: Sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods* *in Natural Language Processing (EMNLP ’02)*, pages 79–86.

Pantel, P. and Lin, D. (1998). SpamCop: A spam classification & organization program. In *Learning for Text Categorization: Papers from the 1998 Workshop*, Madison, Wisconsin.

Paolacci, G., Chandler, J., and Ipeirotis, P. G. (2010). Running experiments on amazon mechanical turk. *Judgment and Decision Making*, 5(5):411–419.

Partee, B. H. (1970). Opacity, coreference, and pronouns. *Synthese*, 21(3):359–385.

Patwardhan, S., Banerjee, S., and Pedersen, T. (2003). Using measures of semantic relatedness for word sense disambiguation. In *Proceedings of the 4th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing ’03)*, pages 241–257.

Pedersen, T., Kulkarni, A., Angheluta, R., Kozareva, Z., and Solorio, T. (2006). An unsupervised language independent method of name discrimination using second order co-occurrence features. In *Proceedings of the 7th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing ’06)*, pages 208–222.

Pedersen, T., Purandare, A., and Kulkarni, A. (2005). Name discrimination by clustering similar contexts. In *Proceedings of the 6th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing ’05)*, pages 226–237.

Peldszus, A. and Stede, M. (2013). From argument diagrams to argumentation mining in texts: A survey. *International Journal of Cognitive Informatics and Natural Intelligence*, 7(1):1–31.

Peng, H., Khashabi, D., and Roth, D. (2015). Solving hard coreference problems. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT ’15)*, pages 809–819.

Pershina, M., He, Y., and Grishman, R. (2015). Personalized page rank for named entity disambiguation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT ’15)*, pages 238–243.

Pilz, A. and Paaß, G. (2011). From names to entities using thematic context distance. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management (CIKM ’11)*, pages 857–866.

Plag, I. (2003). *Word-Formation in English.* Cambridge Textbooks in Linguistics. Cambridge University Press.

Platt, J. C. (1998). Fast training of support vector machines using sequential minimal optimization. In Schölkopf, B., Burges, C. J. C., and Smola, A. J., editors, *Advances in Kernel Methods - Support Vector Learning*, pages 185–208.

Poesio, M., Mehta, R., Maroudas, A., and Hitzeman, J. (2004). Learning to resolve bridging references. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL ’04)*.

Poesio, M., Stuckardt, R., and Versley, Y. (2016). *Anaphora Resolution: Algorithms, Resources, and Applications.* Springer Publishing Company, Incorporated, 1st edition.

Popescu, A.-M. and Etzioni, O. (2005). Extracting product features and opinions from reviews.

In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT-EMNLP '05)*, pages 339–346.

Postal, P. M. (1968). Cross-over phenomena: A study in the grammar of coreference. Research report, Thomas J. Watson Research Center, IBM, Yorktown Heights, N.Y.

Postma, M., Ilievski, F., Vossen, P., and van Erp, M. (2016). Moving away from semantic overfitting in disambiguation datasets. In *Proceedings of the Workshop on Uphill Battles in Language Processing: Scaling Early Achievements to Robust Methods*, pages 17–21.

Pradhan, S., Moschitti, A., Xue, N., Uryupina, O., and Zhang, Y. (2012). Conll-2012 shared task: Modeling multilingual unrestricted coreference in ontoNotes. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL '12)*, pages 1–40.

Pradhan, S., Ramshaw, L., Marcus, M., Palmer, M., Weischedel, R., and Xue, N. (2011). Conll-2011 shared task: Modeling unrestricted coreference in ontoNotes. In *Proceedings of the 15th Conference on Computational Natural Language Learning (CoNLL 2011)*, Portland, Oregon.

Pradhan, S. S., Hovy, E., Marcus, M., Palmer, M., Ramshaw, L., and Weischedel, R. (2007). OntoNotes: A unified relational semantic representation. In *Proceedings of the 1st International Conference on Semantic Computing (ICSC '07)*, pages 517–526.

Qin, B., Zhao, Y., Gao, L., and Liu, T. (2008). Recommended or not? give advice on online products. In *Proceedings of the 5th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD '08)*, pages 208–212.

Qiu, G., Liu, B., Bu, J., and Chen, C. (2009). Expanding domain sentiment lexicon through double propagation. In *Proceedings of the 21st International Jont Conference on Artifical Intelligence (IJCAI '09)*, pages 1199–1204.

Quinlan, J. R. (1983). Learning efficient classification procedures and their application to chess end games. In Michalski, R. S., Carbonell, J. G., and Mitchell, T. M., editors, *Machine Learning: An Artificial Intelligence Approach*, pages 463–482. Springer, Berlin, Heidelberg.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, 1(1):81–106.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Raaijmakers, S., Truong, K., and Wilson, T. (2008). Multimodal subjectivity analysis of multiparty conversation. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP '08)*, pages 466–474.

Raghunathan, K., Lee, H., Rangarajan, S., Chambers, N., Surdeanu, M., Jurafsky, D., and Manning, C. (2010). A multi-pass sieve for coreference resolution. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP '10)*, pages 492–501.

Rahman, A. and Ng, V. (2009). Supervised models for coreference resolution. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP '09)*, pages 968–977.

Rahman, A. and Ng, V. (2010). Inducing fine-grained semantic classes via hierarchical and collective classification. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING '10)*, pages 931–939.

Rahman, A. and Ng, V. (2012). Resolving complex cases of definite pronouns: The winograd schema challenge. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL '12)*, pages 777–789.

Raiman, J. and Raiman, O. (2018). Deeptype: Multilingual entity linking by neural type system evolution. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5406–5413.

Rao, D., McNamee, P., and Dredze, M. (2010). Streaming cross document entity coreference resolution. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters (COLING '10)*, pages 1050–1058.

Rao, D., Mcnamee, P., and Dredze, M. (2013). Entity linking: Finding extracted entities in a knowledge base. In Poibeau, T., Saggion, H., Piskorski, J., and Yangarber, R., editors, *Multi-source, Multilingual Information Extraction and Summarization*, chapter 5, pages 93–116. Springer-Verlag, Berlin Heidelberg.

Ratinov, L. and Roth, D. (2009). Design challenges and misconceptions in named entity recognition. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL '09)*, pages 147–155.

Ratinov, L., Roth, D., Downey, D., and Anderson, M. (2011). Local and global algorithms for disambiguation to wikipedia. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT '11)*, pages 1375–1384.

Raut, V. B. and Londhe, D. D. (2014). Opinion mining and summarization of hotel reviews. In *Proceedings of the 2014 International Conference on Computational Intelligence and Communication Networks (CICN '14)*, pages 556–559.

Rebholz-Schuhmann, D., Jimeno-Yepes, A., van Mulligen, E. M., Kang, N., Kors, J. A., Milward, D., Corbett, P. T., Buyko, E., Beisswanger, E., and Hahn, U. (2010a). CALBC silver standard corpus. *Journal of Bioinformatics and Computational Biology*, 8(1):163–179.

Rebholz-Schuhmann, D., Jimeno-Yepes, A. J., van Mulligen, E. M., Kang, N., Kors, J. A., Milward, D., Corbett, P. T., Buyko, E., Tomanek, K., Beisswanger, E., and Hahn, U. (2010b). The CALBC silver standard corpus for biomedical named entities - A study in harmonizing the contributions from four independent named entity taggers. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC '10)*.

Recasens, M., de Marneffe, M.-C., and Potts, C. (2013). The life and death of discourse entities:

Identifying singleton mentions. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT '13)*, pages 627–633.

Recasens, M. and Hovy, E. (2011). Blanc: Implementing the rand index for coreference evaluation. *Natural Language Engineering*, 17(04):485–510.

Recasens, M., Màrquez, L., Sapena, E., Martí, M. A., Taulé, M., Hoste, V., Poesio, M., and Versley, Y. (2010). Semeval-2010 task 1: Coreference resolution in multiple languages. In *Proceedings of the 5th International Workshop on Semantic Evaluation (SemEval '10)*, pages 1–8.

Rees, A. M. and Schultz, D. G. (1967). *A field experiment approach to the study of relevance assessments in relation to document searching*, volume 2. Center for Documentation and Communication Research, School of Library Science, Case Western Reserve University, Cleveland, OH, USA.

Riloff, E. and Wiebe, J. (2003). Learning extraction patterns for subjective expressions. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing (EMNLP '03)*, pages 105–112.

Ritter, A., Clark, S., Mausam, and Etzioni, O. (2011). Named entity recognition in tweets: An experimental study. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP '11)*, pages 1524–1534.

Rosen-Zvi, M., Griffiths, T., Steyvers, M., and Smyth, P. (2004). The author-topic model for authors and documents. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence (UAI '04)*, pages 487–494.

Rosenthal, S., Farra, N., and Nakov, P. (2017). Semeval-2017 task 4: Sentiment analysis in twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval '17)*, pages 502–518. Association for Computational Linguistics.

Rosenthal, S. and McKeown, K. R. (2012). Detecting opinionated claims in online discussions. In *Sixth IEEE International Conference on Semantic Computing (ICSC '12)*, pages 30–37.

Rosenthal, S., Nakov, P., Kiritchenko, S., Mohammad, S., Ritter, A., and Stoyanov, V. (2015). Semeval-2015 task 10: Sentiment analysis in twitter. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval '15)*, pages 451–463.

Rösiger, I., Köper, M., Nguyen, K. A., and Schulte im Walde, S. (2018a). Integrating predictions from neural-network relation classifiers into coreference and bridging resolution. In *Proceedings of the 1st Workshop on Computational Models of Reference, Anaphora and Coreference*, pages 44–49.

Rösiger, I., Riester, A., and Kuhn, J. (2018b). Bridging resolution: Task definition, corpus resources and rule-based experiments. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING '18)*, pages 3516–3528.

Ross, J., Irani, L., Silberman, M. S., Zaldivar, A., and Tomlinson, B. (2010). Who are the crowdworkers?: Shifting demographics in mechanical turk. In *Extended Abstracts on Human*

Factors in Computing Systems, pages 2863–2872.

Rudrapal, D., Jamatia, A., Chackma, K., Das, A., and Gambäck, B. (2015). Sentence boundary detection for social media text. In *Proceedings of the 12th International Conference on Natural Language Processing*, pages 91–97.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning Representations by Back-propagating Errors. *Nature*, 323(6088):533–536.

Rustamov, S., Mustafayev, E., and Clements, M. (2013). Sentence-level subjectivity detection using neuro-fuzzy models. In *Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 108–114.

Röder, M., Usbeck, R., Hellmann, S., Gerber, D., and andreas Both (2014). N^3 - A collection of datasets for named entity recognition and disambiguation in the nlp interchange format. In *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC '14)*.

Santorini, B. (1990). Part-Of-Speech tagging guidelines for the Penn Treebank project (3rd revision, 2nd printing). Technical report, Department of Linguistics, University of Pennsylvania, Philadelphia, PA, USA.

Sarmento, L., Kehlenbeck, A., Oliveira, E., and Ungar, L. (2009). An approach to web-scale named-entity disambiguation. In *Proceedings of the 6th International Conference on Machine Learning and Data Mining in Pattern Recognition (MLDM '09)*, pages 689–703.

Sarmento, L., Pinto, A. S., and Cabral, L. (2006). REPENTINO - A wide-scope gazetteer for entity recognition in portuguese. In Vieira, R., Quaresma, P., Nunes, M. d. G. V., Mamede, N. J., Oliveira, C., and Dias, M. C., editors, *Computational Processing of the Portuguese Language*, volume 3960 of *Lecture Notes in Computer Science*, pages 31–40. Springer, Berlin, Heidelberg.

Scheible, C., Laws, F., Michelbacher, L., and Schütze, H. (2010). Sentiment translation through multi-edge graphs. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters (COLING '10)*, pages 1104–1112.

Schmid, H. (1994). Probabilistic part-of-speech tagging using decision trees. In *Proceedings of the International Conference on New Methods in Language Processing*, Manchester, UK.

Schmid, H. (1995). Improvements in part-of-speech tagging with an application to german. In *Proceedings of the ACL SIGDAT-Workshop*, pages 47–50.

Schütze, H. (1992). Context space. In *Working Notes of the AAAI Fall Symposium on Probabilistic Approaches to Natural Language*, pages 113–120.

Sen, P. (2012). Collective context-aware topic models for entity disambiguation. In *Proceedings of the 21st International Conference on World Wide Web (WWW '12)*, pages 729–738.

Severyn, A. and Moschitti, A. (2015). Twitter sentiment analysis with deep convolutional neural networks. In *Proceedings of the 38th Annual International ACM SIGIR Conference on Research and Development on Information Retrieval (SIGIR '15)*, pages 959–962.

Shen, W., Wang, J., and Han, J. (2015). Entity linking with a knowledge base: Issues, techniques, and solutions. *Knowledge and Data Engineering*, 27(2):443–460.

Shen, W., Wang, J., Luo, P., and Wang, M. (2012). Linden: linking named entities with knowledge base via semantic knowledge. In *Proceedings of the 21st International Conference on World Wide Web (WWW '12)*, pages 449–458.

Shen, W., Wang, J., Luo, P., and Wang, M. (2013). Linking named entities in tweets with knowledge base via user interest modeling. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '13)*, pages 68–76.

Shu, L., Long, B., and Meng, W. (2009). A latent topic model for complete entity resolution. In *Proceedings of the 2009 IEEE International Conference on Data Engineering (ICDE '09)*, pages 880–891.

Sil, A., Kundu, G., Florian, R., and Hamza, W. (2017). Neural cross-lingual entity linking.

Singh, S., Subramanya, A., Pereira, F., and McCallum, A. (2011). Large-scale cross-document coreference using distributed inference and hierarchical models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT '11)*, pages 793–803.

Snow, R., O'Connor, B., Jurafsky, D., and Ng, A. Y. (2008). Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP '08)*, pages 254–263.

Socher, R., Pennington, J., Huang, E. H., Ng, A. Y., and Manning, C. D. (2011). Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP '11)*, pages 151–161.

Song, Y., Jiang, J., Zhao, W. X., Li, S., and Wang, H. (2012). Joint learning for coreference resolution with markov logic. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL '12)*, pages 1245–1254.

Soon, W. M., Ng, H. T., and Lim, D. C. Y. (2001). A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27(4):521–544.

Sorokin, A. and Forsyth, D. (2008). Utility data annotation with amazon mechanical turk. In *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–8.

Sproat, R., Black, A. W., Chen, S., Kumar, S., Ostendorf, M., and Richards, C. (2001). Normalization of non-standard words. *Computer Speech and Language*, 15(3):287–333.

Staffelbach, M., Sempolinski, P., Hachen, D., Kareem, A., Kijewski-Correa, T., Thain, D., Wei, D., and Madey, G. (2014). Lessons learned from an experiment in crowdsourcing complex citizen engineering tasks with amazon mechanical turk. *CoRR*, abs/1406.7588.

Stede, M., Schneider, J., and Hirst, G. (2018). *Argumentation Mining*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Sun, Y., Lin, L., Tang, D., Yang, N., Ji, Z., and Wang, X. (2015). Modeling mention, context and entity with neural networks for entity disambiguation. In *Proceedings of the 24th International Conference on Artificial Intelligence (IJCAI'15)*, pages 1333–1339.

Sundheim, B. M. (1995). Overview of results of the MUC-6 evaluation. In *Proceedings of the 6th Conference on Message Understanding (MUC6 '95)*, pages 13–31.

Täckström, O. and McDonald, R. (2011). Semi-supervised latent variable models for sentence-level sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers (ACL-HLT '11)*, pages 569–574.

Tafreshi, S. and Diab, M. (2018). Sentence and clause level emotion annotation, detection, and classification in a multi-genre corpus. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018).*

Tan, L. I., Phang, W. S., Chin, K. O., and Anthony, P. (2015). Rule-based sentiment analysis for financial news. In *2015 IEEE International Conference on Systems, Man, and Cybernetics*, pages 1601–1606.

Tang, R., Shaw, Jr., W. M., and Vevea, J. L. (1999). Towards the identification of the optimal number of relevance categories. *Journal of the Association for Information Science and Technology*, 50(3):254–264.

Teufel, S. and van Halteren, H. (2004). Evaluating information content by factoid analysis: Human annotation and stability. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP '04)*, pages 419–426.

Tombros, A. and Sanderson, M. (1998). Advantages of query biased summaries in information retrieval. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development on Information Retrieval (SIGIR '98)*, pages 2–10.

Tong, R. M. (2001). An operational system for detecting and tracking opinions in on-line discussion. In *Proceedings of SIGIR Workshop on Operational Text Classification.*

Toutanova, K., Klein, D., Manning, C. D., and Singer, Y. (2003). Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (NAACL '03)*, pages 173–180.

Tsai, C.-T. and Roth, D. (2016). Cross-lingual wikification using multilingual embeddings. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT '16)*, pages 589–598.

Turney, P. D. (2002). Thumbs up or thumbs down?: Semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL '02)*, pages 417–424.

Turney, P. D. and Littman, M. L. (2003). Measuring praise and criticism: Inference of semantic orientation from association. *ACM Transactions on Information Systems*, 21(4):315–346.

van Erp, M., Mendes, P., Paulheim, H., Ilievski, F., Plu, J., Rizzo, G., and Waitelonis, J. (2016). Evaluating entity linking: An analysis of current benchmark datasets and a roadmap for doing a better job. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC '16)*, pages 4373–4379.

van Halteren, H. and Teufel, S. (2003). Examining the consensus between human summaries: Initial experiments with factoid analysis. In *Proceedings of the HLT-NAACL 03 on Text Summarization Workshop - Volume 5 (HLT-NAACL-DUC '03)*, pages 57–64.

Vapnik, V. (1982). *Estimation of Dependences Based on Empirical Data: Springer Series in Statistics (Springer Series in Statistics)*. Springer, Berlin, Heidelberg.

Vilain, M., Burger, J., Aberdeen, J., Connolly, D., and Hirschman, L. (1995). A model-theoretic coreference scoring scheme. In *Proceedings of the 6th Conference on Message Understanding (MUC6 '95)*, pages 45–52.

Villalba, M. P. G. and Saint-Dizier, P. (2012). Some facets of argument mining for opinion analysis. *COMMA*, 245:23–34.

Vural, V. and Dy, J. G. (2004). A hierarchical method for multi-class support vector machines. In *Proceedings of the 21st International Conference on Machine Learning (ICML '04)*, pages 105–112.

Vuurens, Jeroen, V. A. P. d. . E. C. (2011). How much spam can you take? an analysis of crowdsourcing results to increase accuracy. In *Proceedings of the ACM SIGIR Workshop on Crowdsourcing for Information Retrieval*.

Wachsmuth, H., Kiesel, J., and Stein, B. (2015). Sentiment flow - a general model of web review argumentation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 601–611.

Wachsmuth, H., Trenkmann, M., Stein, B., and Engels, G. (2014). Modeling review argumentation for robust sentiment analysis. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 553–564.

Wang, B. and Wang, H. (2008). Bootstrapping both product features and opinion words from chinese customer reviews with cross-inducing. In *Proceedings of the 3rd International Joint Conference on Natural Language Processing (IJCNLP '08)*.

Wang, Y., Huang, M., Zhu, X., and Zhao, L. (2016). Attention-based LSTM for aspect-level sentiment classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP '16)*, pages 606–615.

Weischedel, R., Palmer, M., Marcus, M., Hovy, E., Pradhan, S., Ramshaw, L., Xue, N., Taylor, A., Kaufman, J., Franchini, M., El-Bachouti, M., Belvin, R., and Houston, A. (2011). OntoNotes Release 4.0 LDC2011T03. DVD. Linguistic Data Consortium, Philadelphia.

Weston, J. and Watkins, C. (1998). Multi-class support vector machines. Technical report CSD-TR-98-04, Department of Computer Science, Royal Holloway University of London.

Wiebe, J. and Riloff, E. (2005). Creating subjective and objective sentence classifiers from unannotated texts. In *Proceedings of the 6th International Conference on Intelligent Text*

Processing and Computational Linguistics (CICLing '05), pages 486–497.

Wilson, T., Wiebe, J., and Hoffmann, P. (2005). Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT-EMNLP '05)*, pages 347–354.

Wilson, T., Wiebe, J., and Hoffmann, P. (2009). Recognizing contextual polarity: An exploration of features for phrase-level sentiment analysis. *Computational Linguistics*, 35(3):399–433.

Winograd, T. (1972). *Understanding Natural Language*. Academic Press, Inc., Orlando, FL, USA.

Xue, W. and Li, T. (2018). Aspect based sentiment analysis with gated convolutional networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2514–2523.

Yamada, I., Shindo, H., Takeda, H., and Takefuji, Y. (2016). Joint learning of the embedding of words and entities for named entity disambiguation. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, pages 250–259.

Yang, B., Cardie, C., and Frazier, P. I. (2015). A hierarchical distance-dependent bayesian model for event coreference resolution. *CoRR*, pages 517–528.

Yang, C., Lin, K. H.-Y., and Chen, H.-H. (2007a). Building emotion lexicon from weblog corpora. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 133–136.

Yang, C., Lin, K. H.-Y., and Chen, H.-H. (2007b). Emotion classification using web blog corpora. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence*, pages 275–278.

Yang, X., Su, J., Zhou, G., and Tan, C. L. (2004). An np-cluster based approach to coreference resolution. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING '04)*.

Yang, X., Zhou, G., Su, J., and Tan, C. L. (2003). Coreference resolution using competition learning approach. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL '03)*, pages 176–183.

Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., and Hovy, E. (2016). Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT '16)*, pages 1480–1489.

Yao, L., Mimno, D., and McCallum, A. (2009). Efficient methods for topic model inference on streaming document collections. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '09)*, pages 937–946.

Yarowsky, D. (1992). Word-sense disambiguation using statistical models of roget's categories trained on large corpora. In *Proceedings of the 14th International Conference on Computational Linguistics (COLING '92)*, pages 454–460.

Yarowsky, D. (1995). Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL '95)*, pages 189–196.

Yessenalina, A. and Cardie, C. (2011). Compositional matrix-space models for sentiment analysis. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP '11)*, pages 172–182.

Yessenalina, A., Yue, Y., and Cardie, C. (2010). Multi-level structured models for document-level sentiment classification. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP '10)*, pages 1046–1056.

Yin, X. and Shah, S. (2010). Building taxonomy of web search intents for name entity queries. In *Proceedings of the 19th International Conference on World Wide Web(WWW '10)*, page 1001–1010.

Yogatama, D., Gillick, D., and Lazic, N. (2015). Embedding methods for fine grained entity type classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing: Short Papers (ACL-IJCNLP '15)*, pages 291–296.

Yosef, M. A., Bauer, S., Hoffart, J., Spaniol, M., and Weikum, G. (2012). HYENA: hierarchical type classification for entity names. In *Proceedings of the 24th International Conference on Computational Linguistics: Posters (COLING '12)*, pages 1361–1370.

Yu, H. and Hatzivassiloglou, V. (2003). Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing (EMNLP '03)*, pages 129–136.

Yu, J., Zha, Z.-J., Wang, M., and Chua, T.-S. (2011). Aspect ranking: Identifying important product aspects from online consumer reviews. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT '11)*, pages 1496–1505.

Yu, X., Rong, W., Zhang, Z., Ouyang, Y., and Xiong, Z. (2019). Multiple level hierarchical network based clause selection for emotion cause extraction. *IEEE Access*, PP:9071–9079.

Zesch, T., Müller, C., and Gurevych, I. (2008). Extracting lexical semantic knowledge from wikipedia and wiktionary. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC '08)*.

Zhang, L. and Liu, B. (2011). Identifying noun product features that imply opinions. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers (ACL-HLT '11)*, pages 575–580.

Zhang, L., Liu, B., Lim, S. H., and O'Brien-Strain, E. (2010). Extracting and ranking product features in opinion documents. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters (COLING '10)*, pages 1462–1470.

Zhang, L., Wu, T., Xu, L., Wang, M., Qi, G., and Sack, H. (2019). Emerging entity discovery using web sources.

Zhang, W., Sim, Y. C., Su, J., and Tan, C. L. (2011). Entity linking with effective acronym expansion, instance selection and topic modeling. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence - Volume Volume Three (IJCAI '11)*, pages 1909–1914.

Zhao, J. and Liu, F. (2008). Product named entity recognition in chinese text. *Language Resources and Evaluation*, 42(2):197–217.

Zheng, Z., Chen, K., Sun, G., and Zha, H. (2007). A regression framework for learning ranking functions using relative relevance judgments. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development on Information Retrieval (SIGIR '07)*, pages 287–294.

Zheng, Z., Li, F., Huang, M., and Zhu, X. (2010). Learning to link entities with knowledge base. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT '10)*, pages 483–491.

Zhou, G. and Su, J. (2002). Named entity recognition using an hmm-based chunk tagger. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL '02)*, pages 473–480.

Zhu, D. and Carterette, B. (2010). An analysis of assessor behavior in crowdsourced preference judgments. In *Proceedings of the ACM SIGIR 2010 Workshop on Crowdsourcing for Search Evaluation (CSE 2010)*, pages 21–26.

Zhuang, L., Jing, F., and Zhu, X.-Y. (2006). Movie review mining and summarization. In *Proceedings of the 15th ACM International Conference on Information and Knowledge Management (CIKM '06)*, pages 43–50.

Zipf, G. K. (1949). Human behavior and the principle of least effort.