Large-Scale Analysis of Textual and Multivariate Data Combining Machine Learning and Visualization

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

Vorgelegt von

Johannes Simon Knittel

aus Böblingen

Hauptberichter:	Prof. Dr. Thomas Ertl
Mitberichter:	Prof. Dr. Shixia Liu
	Prof. Dr. Andreas Kerren
Tag der mündlichen Prüfung:	02.05.2022

Institut für Visualisierung und Interaktive Systeme
der Universität Stuttgart

2022

Contents

List of Figures

Chapter 2

Chapter 3

Chapter 4

Chapter 5

Chapter 6

Chapter 7

List of Tables

Chapter 3

Chapter 4

Chapter 5

Chapter 7

List of Abbreviations and Acronyms

AI	artificial intelligence
AIX	artificial intelligence for visually explaining data sets
CPU	central processing unit
GPGPU	general purpose computating on graphics processing units
GPU	graphics processing unit
KDD	knowledge discovery in databases
ML	machine learning
NMI	normalized mutual information
skMeans	Spherical k-Means
VA	visual analytics
XAI	explainable artificial intelligence

Acknowledgments

I am very grateful for the support of several people that helped me a lot in conducting research and in writing this dissertation. First of all, I would like to thank my advisor Thomas Ertl very much for giving me the opportunity to work on exciting projects for my doctoral degree, for continuously supporting and advising me, as well as for providing me with funding to attend several conferences around the world. I also want to thank Steffen Koch for his great help and support throughout the years, for fruitful scientific discussions, and for collaborating with me on many papers.

It would not have been possible to write this dissertation without the funding from the Deutsche Forschungsgesellschaft (German Research Foundation) as part of the projects VAOST, VA4VGI, and CRETA. I am very happy that Shixia Liu agreed to review my dissertation and I really enjoyed our discussions on joint paper projects. I want to extend my gratitude to Andreas Kerren for taking the time to review my dissertation, for his helpful advice, and for traveling all the way from Sweden to attend my oral defense.

There are always ups and downs when working towards a doctoral degree, but my wonderful colleagues at VIS and VISUS made sure that it was not only hard work but also fun work. In particular, I want to thank Franziska Huth for introducing me to bouldering and my colleagues Cristina Morariu, Gleb Tkachev, Max Franke, Moritz Knabben, Tanja Blascheck, Andrés Lalama, Valentin Bruder, Alexander Straub, David Hägele, Moataz Abdelaal, Michael Becher, Hermann Pflüger, and Kuno Kurzhals for profound but also amusing discussions, game nights, and joint activities. I could always count on the support of Margot Roubicek, for which I am grateful.

Many thanks also go to my good friend Dominique Rau who has always encouraged me and inspired me in many ways. Finally, my deepest thanks go to my parents, my brother, and my partner for their endless love and for always having my back.

Abstract

In a mostly digitized world, we collect, store, process, and make use of massive amounts of data every day. Data has long become an asset for many stakeholders since it promises to hold valuable information and may ultimately lead to new knowledge. However, having access to huge data sets does not necessarily translate to gaining insights from it. We need methods that facilitate the analysis of such data sets to achieve this.

The design, scope, and specification of existing analysis methods vary greatly. For instance, if we can express the analysis goal in a formal and machine-interpretable way, we may apply statistics, machine learning approaches, or simple database operations to learn from or gain insights into data sets. In the case of more complex problem queries or explorative goals, though, it is often infeasible or insufficient to apply such straightforward methods. Visual analytics approaches gained traction in recent years to tackle this challenge. Combining interactive data analytics with human intelligence bears the potential of revealing insights that both machines and human beings would not achieve on their own.

With the rise of social media and its profound impact on society, the amount of published content on a daily basis has vastly increased, prompting an increased need for scalable solutions to analyze large sets of posts and monitor current developments in real-time. A similar case can be made for the visual analysis of data sets in health care, industries, and science due to the rising number of digital processes and measurements. This thesis aims to enable such large-scale visual analyses and has a particular focus on text and multivariate data since they represent a large proportion of relevant big data sets in the field of visual analytics.

On the one hand, it is particularly intriguing to analyze larger data sets as they may exhibit interesting but rare or subtle patterns and relationships which would not be apparent on smaller subsets of the data. Additionally, many data points help to reduce the uncertainty of findings. On the other hand, it is also particularly challenging to develop approaches that support interactive analyses on such big data sets. For multivariate data sets with many dimensions, the set of potential variable groupings that could exhibit an interesting pattern grows prohibitively large. This poses a problem for both efficient data analytics and effective visual mappings. Similar problems arise when analyzing large text collections. Previous approaches have rarely been developed for and tested with data set sizes that exceed tens of thousands of documents. Additionally, few approaches natively support the online analysis of streaming data due to the algorithmic constraints and increased conceptual complexity this would entail.

To tackle these challenges, this thesis introduces the concept of leveraging artificial intelligence for visually explaining large data sets efficiently and effectively. If we train a machine learning model on a data set, we can use that model for computing predictions on novel data items, but we may also discover interesting relationships in the training set by visually analyzing the model itself. Similarly, clustering algorithms help to structure (unlabeled) input data, but the analysis of the resulting clusters may also reveal characteristics of the underlying data set. Based on this concept, this work presents several approaches for analyzing textual and multivariate data.

Visual text analysis belongs to one of the key target domains in the field of visual analytics as it is generally difficult for machines to analyze and interpret text due to its unstructured nature. This thesis introduces a technique for analysts to interactively explore large document collections based on a novel visual encoding that conveys both temporal and contextual information. It further proposes a comprehensive system for the real-time analysis of streaming social media posts that is based on an efficient dynamic clustering algorithm.

Text first hast to be converted into a suitable vector representation. When analyzing multivariate data set, however, we can utilize the tabular structure imposed by the already given attributes. Each data item corresponds to an array of values, which makes subsequent computations and aggregations more straightforward. However, previous visual analytics approaches either focus on simpler (e.g., linear) models, or do not scale well with the number of attributes. Hence, this thesis proposes an efficient neural network-based technique for the visual analysis of multivariate data sets that is able to recognize more complex relationships but also scales well to many input dimensions.

Despite continuous advancements in computing power, gaining insights into large data sets remains one of the greatest challenges today. This work proposes several techniques to facilitate such large-scale analyses of textual and multivariate data, but it also aims to incite and steer new visual analytics approaches that are both scalable and effective.

German Abstract
—Zusammenfassung—

In einer weitgehend digitalisierten Welt erheben, speichern, verarbeiten und nutzen wir täglich riesige Datenmengen. Daten sind für viele Stakeholder längst zu einer Währung geworden, da man sich davon wertvolle Informationen verspricht und sie letztendlich zu neuem Wissen führen können. Der bloße Zugang zu riesigen Datensätze bedeutet jedoch nicht unbedingt, dass man daraus auch Erkenntnisse gewinnen kann. Dazu brauchen wir Methoden, die die Analyse solcher Datensätze ermöglichen.

Design, Umfang und Spezifikation bestehender Analysemethoden variieren stark. Wenn man beispielsweise das Analyseziel mathematisch oder maschinell interpretierbar ausdrücken kann, könnte man Statistiken, Ansätze des maschinellen Lernens oder einfache Datenbankoperationen anwenden, um aus Datensätzen zu lernen oder Einblicke in diese zu gewinnen. Bei komplexeren Problemfragen oder explorativen Zielen ist es jedoch oft nicht möglich oder nicht ausreichend, solche simpleren Methoden anzuwenden. Visual Analytics-Ansätze haben in den letzten Jahren an Bedeutung gewonnen, um dieser Herausforderung zu begegnen. Die Kombination von interaktiver Datenanalyse mit menschlicher Intelligenz birgt das Potenzial, Erkenntnisse zu gewinnen, die sowohl Maschinen als auch Menschen allein nicht erreichen würden.

Mit dem Aufstieg der sozialen Medien und ihren tiefgreifenden Auswirkungen auf die Gesellschaft hat die Menge der täglich veröffentlichten Inhalte stark zugenommen, was zu einem erhöhten Bedarf an skalierbaren Lösungen führt, um große Mengen von Beiträgen zu analysieren und aktuelle Entwicklungen in Echtzeit zu überwachen. Ähnliches gilt für die visuelle Analyse von Datensätzen im Gesundheitswesen, in der Industrie und in der Wissenschaft aufgrund der steigenden Anzahl digitaler Prozesse und Messungen. Diese Arbeit zielt darauf ab, solche groß angelegten visuellen Analysen zu ermöglichen, und legt einen besonderen Fokus auf textuelle und multivariate Daten, da sie einen signifikanten Anteil solcher ‘Big Data’ Problemstellungen im Bereich Visual Analytics ausmachen.

Einerseits ist es besonders interessant, größere Datensätze zu analysieren, da sie bedeutende, aber seltene oder subtile Muster und Beziehungen aufweisen können, die bei kleineren Teilmengen der Daten nicht sichtbar wären. Darüber hinaus tragen viele Datenpunkte aber auch dazu bei, die Unsicherheit von Analyseergebnissen zu reduzieren. Andererseits ist es aber auch besonders anspruchsvoll, Ansätze zu entwickeln, die interaktive Analysen von solchen großen Datenmengen unterstützen. Bei multivariaten Datensätzen mit vielen Dimensionen wird die Menge potenzieller Variablengruppierungen, die ein interessantes Muster aufweisen könnten, enorm groß. Dies stellt sowohl für eine effiziente Datenanalyse als auch für effektive visuelle Mappings ein Problem dar. Ähnliche Herausforderungen treten bei der Analyse großer Textsammlungen auf. Bisherige Ansätze wurden selten für Datensatzgrößen von deutlich mehr als Zehntausenden von Dokumenten entwickelt und getestet. Darüber hinaus unterstützen nur wenige Ansätze nativ die Online-Analyse von Streaming-Daten aufgrund der algorithmischen Einschränkungen und der erhöhten konzeptionellen Komplexität, was dies mit sich bringen würde.

Um diese Herausforderungen anzugehen, stellt diese Arbeit das Konzept vor, Methoden der künstlichen Intelligenz auszunutzen, um große Datenmengen effizient und effektiv visuell zu erklären. Wenn man ein Machine Learning Modell an einem Datensatz trainiert, kann man dieses Modell verwenden, um für weitere Datenpunkte Vorhersagen zu berechnen. Man kann jedoch möglicherweise auch interessante Beziehungen im zugrundeliegenden Trainingsdatensatz entdecken, indem man das trainierte Modell an sich visuell analysiert. Beispielsweise können Clustering-Algorithmen helfen, (nicht gelabelte) Eingabedaten zu strukturieren, aber die Analyse der resultierenden Cluster kann auch Merkmale des zugrundeliegenden Datensatzes aufdecken. Basierend auf diesem Konzept werden in dieser Arbeit verschiedene Ansätze zur Analyse textueller und multivariater Daten vorgestellt.

Die visuelle Textanalyse gehört zu den wichtigsten Anwendungsbereichen von Visual Analytics, da es für Computer generell schwierig ist, Texte zu analysieren und zu interpretieren aufgrund der unstrukturierten Datenbeschaffenheit. Diese Dissertation stellt eine Technik für Analysten vor, um große Dokumentensammlungen interaktiv zu erkunden. Sie basiert auf einer neuartigen visuellen Kodierung, die sowohl zeitliche als auch kontextbezogene Informationen vermittelt. Darüber hinaus wird ein umfassendes System für die Echtzeitanalyse von kontinuierlich geteilten Social-Media-Posts vorgeschlagen, das auf einem effizienten dynamischen Clustering-Algorithmus basiert.

Während Text erst in eine vektorbasierte Repräsentation umgewandelt werden muss, kann man bei der Analyse multivariater Datensätze die durch die bereits vorgegebenen Attribute tabellarische Struktur ausnutzen. Jedes Datenelement entspricht einem Array von Werten, was nachfolgende Berechnungen und Aggregationen einfacher macht. Bisherige Ansätze zur visuellen Analyse konzentrieren sich jedoch entweder auf einfachere (z.B. lineare) Modelle oder skalieren nicht gut mit der Anzahl der Attribute. Daher schlägt diese Dissertation eine effiziente, auf neuronalen Netzen basierende Technik zur visuellen Analyse von multivariaten Datensätzen vor, die in der Lage ist, komplexere

Zusammenhänge zu erkennen, aber auch gut mit der Anzahl der Eingabedimensionen skaliert.

Trotz ständiger Fortschritte in der Rechenleistung bleibt es auch heute eine große Herausforderung, Einblicke in große Datensätze zu erlangen und Erkenntnisse daraus zu gewinnen. Diese Arbeit schlägt verschiedene Techniken vor, um solche groß angelegten Analysen von textuellen und multivariaten Daten zu erleichtern. Sie zielt aber auch darauf ab, die Erforschung neuer Visual Analytics Ansätze gezielt anzuregen, die sowohl skalierbar als auch mächtig sind.

CHAPTER 1

Introduction

In a mostly digitized world, we collect, store, process, and make use of massive amounts of data every day. Data has long become an asset for many stakeholders since it promises to hold valuable information and may ultimately lead to new knowledge. However, having access to huge data sets does not necessarily translate to gaining insights from it. We need methods that facilitate the analysis of such data sets to achieve this.

Numerous approaches have been developed in the past decades that differ greatly regarding their target domain, target audience, supported input data, features, scope of analysis, specificity, requirements, ease of use, and efficiency. The field of visual analytics (Thomas and Cook [2005]; Keim et al. [2010a]) has gained traction in recent years to tackle a specific set of cases that involve more complex problem queries or explorative goals. According to Thomas and Cook [2005], one major goal of visual analytics is to "detect the expected and discover the unexpected". It is sometimes challenging or even impossible to precisely formulate the goal of the analysis beforehand. For instance, given a set of news reports, we might want to find out which major events and topics were discussed to which extent, but this is a broad and vaguely defined task that would be difficult to solve in a generic and automated way. Combining interactive data analytics with human intelligence, though, bears the potential of revealing insights that both machines and human beings would not achieve on their own.

With the rise of social media and its profound impact on society, the amount of published content on a daily basis has vastly increased, prompting an increased need for scalable solutions to analyze large sets of posts and monitor current developments in real-time. A similar case can be made for the visual analysis of data sets in health care, industries, and science due to the rising number of digital processes and measurements. This thesis aims to enable such large-scale visual analyses and has a particular focus on text and multivariate data since they represent a large proportion of relevant big data sets in the field of visual analytics.

In the case of multivariate datasets, the attributes are already given, while text must first be converted into a suitable vector representation. Visual text analytics approaches often focus on analyzing the textual content of individual documents or document collections, whereas approaches for exploring multivariate datasets typically aim to find and visualize relationships between attributes and sets of attributes. The transitions are fluid, though. For instance, text documents often contain additional structured data that we may want to include in our analysis (e.g., publishing date of an article). Multivariate data can also contain unstructured content that may be relevant to the task at hand (e.g., text input in survey responses). Furthermore, analysts may wish to find and understand relationships in documents after they have been processed, for example, to examine correlations between extracted entities or understand the evolution of topics over time.

Analyzing larger data sets is particularly intriguing for several reasons. They may exhibit interesting complex or low-key patterns and relationships which cannot be inferred from smaller subsets of the data. For instance, Ferwerda et al. [2020] estimate that, for OECD countries, money laundering amounts to about two percent of the gross domestic product (GDP). A compact random sample of all transactions would therefore likely include close to none of such dubious transactions (assuming similar distributions of transaction sizes), so the analysis of this sample would not lead to meaningful insights regarding money laundering. Another advantage of big data sets is that, even in the case of larger effect sizes, many data points help to reduce the uncertainty of findings.

1.1 Problem Statement

Comprehensive and timely analyses of big document collections such as news reports, business reports, and social media posts are of vital interest for many stakeholders. Several visual text mining and analysis approaches have thus been proposed to serve the needs of, for instance, business analysts, journalists, traders, and intelligence officials.

On the one hand, the field of visual analytics seems to be well suited for making sense of documents and document collections due to the unstructured nature of the data. On the other hand, though, providing an interactive experience significantly constrains the complexity and duration of on-the-fly computations. This partly explains why previous approaches have rarely been developed for and tested with data set sizes that exceed tens of thousands of documents, even though analyzing massive amounts of data is one of the core goals of the field (Thomas and Cook [2005]; Keim et al. [2010a]).

Furthermore, only few approaches natively support real-time analyses of streaming documents. The conceptual complexity increases because we lose the global view of the entire data set, data has to be processed with fast algorithms that support incremental updates (Rohrdantz et al. [2011]), and dynamic visualizations have to be developed that preserve the mental map of users (Krstajić and Keim [2013]). There are numerous circumstances that require or would benefit from real-time analyses, though, for instance, if one needs to monitor an ongoing event for reporting or targeted interventions, or if storing all received data for a subsequent offline analysis is impractical.

One advantage of analyzing multivariate data instead of textual data is that we can utilize the tabular structure imposed by the already given attributes. Each data item corresponds to an array of values, which makes subsequent computations and aggregations more straightforward. However, similar issues can be observed regarding the scalability of existing methods. The vastly increasing number of digital processes and measurements in many domains has lead to bigger data sets not only in terms of the number of data items but also in terms of the complexity of items. With many dimensions, the set of potential variable groupings that could exhibit an interesting pattern grows prohibitively large.

This poses a problem in several ways for large-scale interactive visual analyses. In an interactive setting, non-linear data analytics methods are usually not efficient enough to be applied to large data set sizes with hundreds of attributes. Additionally, it is challenging to visualize multivariate data sets effectively with an increasing number of dimensions. As a result, most visual analytics approaches either apply simple (e.g., linear) statistical models to filter, rank and visually aggregate multivariate data sets, or they focus on data sets with less dimensions.

One can argue a lot about what the terms *big* or *large* actually mean; it surely depends on the context and is partially subjective. In this thesis, *big* or *large* refer to sizes that significantly exceed the average norm of what current approaches in the respective field are usually tested on. For instance, in the case of visualizing multivariate data with parallel coordinates, a data set with more than ten dimensions can be considered large because these plots rarely contain more than ten axes due to the limits of human perception. These terms describe relative sizes and, thus, have to be interpreted according to their historical context. They may change their meaning over time in terms of absolute numbers since the average test data set sizes usually also increase over time. For visual document analyses, collections with well over ten thousand documents are therefore called large. For visual multivariate analyses, this applies to data sets with well over ten thousand items or well over ten dimensions.

1.2 Research Questions

Given the outlined research gap in visual text and multivariate analysis, the main research question that guides all topics of this thesis is:

How can we develop more powerful but also more efficient approaches to scale the visual analysis of textual and multivariate data?

This is a challenging endeavor that encompasses many different facets. Hence, this thesis focuses on tackling the following more specific research questions:

- **RQ 1:** How can we visually aggregate text collections efficiently while still providing sufficient context?
- **RQ 2:** How do we facilitate the interactive exploration of large document collections, incorporating both temporal aspects and semantic context?
- **RQ 3:** How can we cluster documents more efficiently regarding the number of clusters and also support dynamic clustering in a streaming setting?
- **RQ 4:** How can we scale the real-time visual analysis of streaming social media posts without relying on specific meta-data or event detection algorithms?
- **RQ 5:** How can we extract and visualize more complex relationships between a dependent variable and potentially hundreds of independent variables in a multivariate data set?

1.3 Contributions

This thesis makes several contributions to scale the visual analysis of textual and multivariate data. *Contribution 6* addresses the main research question of developing both powerful and efficient approaches to scale the visual analysis of textual and multivariate data. Contributions 1–5 map to the respective research question.

- **Contribution 1a:** Section 3.1 proposes *ELSKE*, an efficient keyphrase extraction algorithm, to briefly summarize large document collections (Knittel et al. [2021b]). It aims to extract not only relevant keywords but also frequently appearing longer phrases that are easier to interpret. It avoids computationally expensive prepocessing such as Part-of-Speech tagging to enable a continuous aggregation of streaming documents at regular intervals.

- **Contribution 1b:** An interactive and hierarchical approach has been developed to extract and visualize frequent textual quotes efficiently (Knittel et al. [2019a,b]), which is described in Section 3.2. Each quote is a sequence of phrases which appear in the respective order. These quotes provide a context-rich but still compact summary of short texts such as tweets. They also help to quantify the extent of certain statements, given that they are similarly phrased.

- **Contribution 2:** In Chapter 4, this work proposes *PyramidTags*, a context-aware, word order-aware, and date-aware multi-term tag map for exploring large document collections (Knittel et al. [2021a]). The position of each tag corresponds to the date range in which it mostly appeared in the collection, frequently co-occurring tags are placed nearby, and the word order of the most important tag pairs is preserved, given that they often appear in a particular lexical order. Analysts can select one or several tags to retrieve relevant documents and learn more about the topic.

- **Contribution 3a:** Several techniques to accelerate the Spherical k-Means algorithm on sparse input data for larger values of k are proposed in Section 5.1, making fine-grained clusterings of large document collections feasible, particularly in interactive or real-time settings (Knittel et al. [2021c]).

- **Contribution 3b:** This work further proposes in Section 5.2 an efficient dynamic clustering algorithm based on Spherical k-Means to enable the clustering of streaming documents at regular intervals (Knittel et al. [2022]).

The number of clusters does not have to be specified beforehand and the algorithm aims to minimize the changes between updates such that a continuous visual representation of the dynamically changing clustering is feasible.

- **Contribution 4:** A comprehensive system to visually analyze streaming social media posts in real-time has been developed (Knittel et al. [2022]), which Chapter 6 introduces. Two dynamic clustering processes handle the incoming posts in parallel and independent from each other. The first, more coarse-grained clustering provides analysts with a continuous topical overview, from which they can select topics of interest they want to monitor. Relevant keyphrases in that selection are continuously extracted and the proposed set visualization helps analysts to infer the coverage and co-occurrence of these phrases. The aim of the second, more fine-grained clustering is to provide a diverse yet digestible stream of representative posts related to such a selection of topics. Analysts can increase the resolution and specificity by diving into the selection of topics.

- **Contribution 5:** This thesis introduces *Visual Neural Decomposition* in Chapter 7 to extract and explain how variables in a multivariate data set influence a specific target variable (Knittel et al. [2021d]). The goal is to understand which variables in which circumstances lead to high target values, even if the data set contains a large number of variables. Based on a neural network architecture and a novel regularization term, the method tries to decompose the problem into a set of cases that are visualized separately, each representing one explanation that leads to high target values. The stacked histograms visualization is proposed to visualize the distribution of the variables within each case in relation to the target values.

- **Contribution 6:** This work proposes in Section 2.5 the generalized concept of exploiting artificial intelligence for visually explaining textual and multivariate data sets (Knittel et al. [2021d]). The idea is to train or build a model on the data set and then to visualize the inner workings of this model in order to learn more about the underlying data set. Thus, it is not about visualizing the outputs of a model but about visualizing the model itself. The challenge is to develop powerful and efficient, but also visually interpretable models to scale the analysis of textual and multivariate data sets.

1.4 Thesis Structure

This thesis is structured as follows. Chapter 2 discusses the foundational concepts and related work upon which this thesis builds. It further introduces the concept of exploiting artificial intelligence for visually explaining textual and multivariate data sets. Chapter 3 introduces efficient methods for aggregating large text collections. The first method extracts keyphrases from document collections efficiently (ELSKE). The second method aims to aggregate collections with short yet context-rich quotes. PyramidTags, a context-aware, word order-aware, and date-aware multi-term tag map for exploring large document collections, is proposed in Chapter 4. Chapter 5 presents several strategies for accelerating the Spherical k-Means algorithm and further introduces an efficient dynamic clustering algorithm. Based on this clustering algorithm, a visual analytics approach for analyzing streaming social media posts in real-time has been developed that is described and evaluated in Chapter 6. Chapter 7 presents Visual Neural Decomposition for discovering and explaining non-linear multivariate relationships. Finally, Chapter 8 summarizes and discusses the contributions of this work and provides an outlook regarding future challenges in the context of this thesis.

Parts of this thesis have already been published in the following publications:

J. Knittel, S. Koch, and T. Ertl. Interactive Hierarchical Quote Extraction for Content Insights. In J. Madeiras Pereira and R. G. Raidou, editors, *Proceedings of the EuroVis 2019 Posters*. The Eurographics Association, 2019a

J. Knittel, S. Koch, and T. Ertl. Pattern-Based Semantic and Temporal Exploration of Social Media Messages. In *Proceedings of the 2019 IEEE Conference on Visual Analytics Science and Technology, VAST 2019*, pages 134–135, 2019b

J. Knittel, S. Koch, and T. Ertl. PyramidTags: Context-, Time- And Word Order-Aware Tag Maps to Explore Large Document Collections. *IEEE Transactions on Visualization and Computer Graphics*, 27(12):4455–4468, 2021a

(continues on the next page)

(Continued)

J. Knittel, A. Lalama, S. Koch, and T. Ertl. Visual Neural Decomposition to Explain Multivariate Data Sets. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1374–1384, 2021d

J. Knittel, S. Koch, T. Tang, W. Chen, Y. Wu, S. Liu, and T. Ertl. Real-Time Visual Analysis of High-Volume Social Media Posts. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):879–889, 2022

J. Knittel, S. Koch, and T. Ertl. ELSKE: Efficient Large-Scale Keyphrase Extraction. In *Proceedings of the 21st ACM Symposium on Document Engineering, DocEng 2021*, New York, NY, USA, 2021b. Association for Computing Machinery

J. Knittel, S. Koch, and T. Ertl. Efficient Sparse Spherical K-Means for Document Clustering. In *Proceedings of the 21st ACM Symposium on Document Engineering, DocEng 2021*, New York, NY, USA, 2021c. Association for Computing Machinery

Chapter 2

Foundations and Concept

Scaling is a complex issue; it is not just about making things bigger or smaller. In nature, mammals come in all different sizes, yet the size and volume of cells typically do not differ much between species – in contrast to the metabolic rate that changes depending on the body size (Savage et al. [2007]). Scaling the visual analysis of textual and multivariate data sets is also not simply about applying the same methods only to larger data sets, even if the computational capacity increases with the size of the set. What works on a small scale does not necessarily work on a large scale, and vice versa. For instance, single documents can be shown in full, whereas larger collections have to be aggregated – but it is also easier to aggregate the content of many documents than summarizing the main content of individual documents. Data sets with few dimensions can be visually mapped to physical dimensions or colors, high-dimensional data sets have to be reduced first or visualized with a different strategy.

This thesis proposes a new model for scaling the visual analysis of textual and multivariate data, which is presented in Section 2.5. The idea is to exploit artificial intelligence for visually explaining data sets by visually inspecting interpretable models that were trained on or fitted to the data. The first part of this chapter discusses relevant methods that are fundamental to the contributions of this thesis. Section 2.1 introduces artificial intelligence and machine learning, Section 2.2 outlines the visual analytics process, Section 2.3 discusses existing approaches for the visual analysis of document collections, and Section 2.4 embeds this thesis in the context of visual multivariate analysis techniques.

2.1 Artificial Intelligence and Machine Learning

The idea of creating things that imitate in some ways 'intelligent' behavior of animals or humans is at least hundreds of years old (Nilsson [2009]), but there is still no consensus on how to properly define *artificial intelligence* (or *intelligence* in general). Legg and Hutter [2007] present some of the numerous definitions that researchers have come up with over the years. The contributions of this thesis mainly focus on *machine learning* (ML) which is considered to be a subset of techniques in the field of artificial intelligence (AI) (Helm et al. [2020]), even though the terms *AI* and *machine learning* are sometimes used interchangeably. Nilsson [2005] states that

> "a machine learns whenever it changes its structure, program, or data (based on its inputs or in response to external information) in such a manner that its expected future performance improves. [...] Machine learning usually refers to the changes in systems that perform tasks associated with artificial intelligence (AI). Such tasks involve recognition, diagnosis, planning, robot control, prediction, etc."

Thus, machine learning is about algorithms that aim to generate algorithms – but in an implicit way by harvesting and modeling data. The type of machine learning algorithm depends on the available training data and can be either classified as supervised, unsupervised, or semi-supervised (Sammut and Webb [2010]). Supervised learning means that the data in the training set is annotated with labels which the machine is supposed to learn (e.g., images with categories as labels). Unsupervised learning does not use any preexisting domain knowledge (e.g., clustering based on data distances). Sometimes, learning happens neither fully supervised nor unsupervised, for instance, if only parts of the data are labeled or the learning takes into account at least some kind of domain knowledge (e.g., interactive clustering). These edge cases are therefore often classified as semi-supervised learning.

Machine learning is a broad field and an adequate introduction would go beyond the scope of this thesis. This section therefore discusses techniques that are important for the contributions of this thesis.

2.1.1 Particle Swarm Optimization

Kennedy and Eberhart [1995] developed an evolutionary algorithm for optimizing continuous non-linear functions, which was later improved by Shi and Eberhart [2002] with an additional hyper-parameter. Given a function

$f : \mathbb{R}^n \to \mathbb{R}$, $\boldsymbol{x} \mapsto f(\boldsymbol{x})$, the optimization objective is to find the point $\boldsymbol{x}$ that minimizes the function value $f(\boldsymbol{x})$ (or is a good local minimum in the case of non-convex functions). The algorithm is partly inspired by the swarm behavior of flying birds and is therefore called *particle swarm optimization* (PSO). One of its advantages is that it is gradient-free as it only needs to evaluate the function at any point. It has been adopted for a wide range of different application domains, including machine learning and modeling (Poli et al. [2007]).

The idea of PSO is that a group of *particles* collectively try to approach the (local) minimum of the function. Each particle i has a position $\boldsymbol{x}_i^t$ that represents its current solution at step t, and a velocity $\boldsymbol{v}_i^{t+1}$ that determines the direction of the update for calculating its position in the next iteration $t+1$. The velocity depends on its previous value, the distance of the particle to the best local solution $\boldsymbol{p}_i$ (the position that has lead to the smallest value of $f(\boldsymbol{x})$ among all of its previous positions), and the distance of the particle to the best group solution $\boldsymbol{p}$ (the position that has lead to the smallest value of $f(\boldsymbol{x})$ among all particles and their previous positions).

In the beginning, each of the M particles is initialized with a random position and velocity (within a certain interval, e.g., $[-1, 1]$). Then, the algorithm loops through the following steps until the maximum number of iterations N is reached:

1. Evaluate $f(\boldsymbol{x}_i)$ on all positions to determine the current *fitness value* of every particle.

2. Check whether any fitness value of a particle i is smaller than the local solution $\boldsymbol{p}_i$ or the global solution $\boldsymbol{p}$, and replace them accordingly if this is the case.

3. Calculate the new velocity for each particle:

$$\boldsymbol{v}_i^{t+1} = W \cdot \boldsymbol{v}_i^t + C_1 \cdot r_1 \cdot (\boldsymbol{p}_i - \boldsymbol{x}_i) + C_2 \cdot r_2 \cdot (\boldsymbol{p} - \boldsymbol{x}_i) \tag{2.1}$$

   where r_1, r_2 are two random values in the range $[0, 1)$, and W, C_1, C_2 (constant) hyper-parameters.

4. Calculate the new position for each particle:

$$\boldsymbol{x}_i^{t+1} = \boldsymbol{x}_i^t + \boldsymbol{v}_i^{t+1} \tag{2.2}$$

Different strategies for selecting the optimal hyper-parameters have been proposed, but $W = 0.7298$, $C_1 = C_2 = 1.49618$, $M < 40$ is a frequently used combination (Poli et al. [2007]; Wang et al. [2018]) that is also adopted in this thesis.

2.1.2 Document Clustering

Automatically grouping text into clusters helps to structure large document collections, which is beneficial for a wide variety of tasks such as document browsing, information retrieval, collection summarization, and classification (Aggarwal and Zhai [2012]; Allahyari et al. [2017]). The underlying assumption of most clustering algorithms is that the items within a group should be similar to each other and dissimilar to every other item (Gentle et al. [1991]). Numerous algorithms have been developed that can be divided into several categories, for instance, which type of structure is produced (flat or hierarchical), which measure for determining group memberships is employed (e.g., distance-based or probabilistic), and whether they are dynamic (online) to work with streams (Aggarwal and Zhai [2012]).

Document Representations

In contrast to quantitative data, text input has to be converted into a suitable vector representation first, which poses several challenges. The vocabulary of a corpus is often huge, yet individual documents such as news reports typically only contain up to hundreds of words. In addition, different encodings generally exhibit different strengths and weaknesses such as human interpretability and the ability to capture subtle semantics. Bag-of-words representations are simple to build and easy to interpret, but more advanced neural network-based document embeddings, for instance, based on *doc2vec* (Le and Mikolov [2014]) or large Transformer-models (Devlin et al. [2019]; Radford et al. [2020]), may represent the semantics better.

What all strategies have in common is that the document must first be tokenized into a sequence of tokens. The exact definition of what constitutes a token varies depending on the specific use case and domain background (Webster and Kit [1992]). This work adopts the frequently used strategy of interpreting single words or numbers as tokens, which are usually delimited by spaces and punctuation marks in western languages.

Applying stemming or lemmatization algorithms can help to unify different variants of the same word. *Stemming* reduces different variants to one common artificial root term (stem), whereas *lemmatization* tries to find the actual base form (lemma) with regard to the context of the sentence (Manning et al. [2008]). However, this work rarely applies either strategy because they suffer from several limitations when applied to the large-scale visual analysis of documents. Stemming is an efficient but also error-prone process. Due to the rule-based nature of stemming, similar words with a different meaning are often reduced to the same stem. Another disadvantage is that the artificial roots are difficult to understand. Lemmatization is a more advanced but also computationally expensive preprocessing step since it requires a proper morphological analysis of each word. Furthermore, reducing each word to its root (artificial or not) makes subsequent analysis tasks dealing with sequences of tokens (i.e., phrases) more difficult to handle and interpret.

For representing documents as vectors, this thesis applies the common TF-IDF weighted bag-of-words approach (Salton and Buckley [1988]), in which each term is mapped to a vector index, and for each present term in the document the corresponding vector entry is set to the respective TF-IDF weight. TF-IDF weights terms x with their frequency in the document $f_s(x)$ in relation to the document frequency of the term in a reference collection $f_d(x)$ comprising N documents to focus on candidates that appear unusually often:

$$\text{TF-IDF}(x) = f_s(x) \ln \frac{N}{f_d(x)} \tag{2.3}$$

A list of stop words that should be ignored is often used to improve the results. While conceptually simple, it has several benefits that are important for the efficient visual analysis of large collections. It is straightforward to integrate new terms which have not been present in the reference corpus, whereas embeddings are usually trained on and tailored to a specific data set. In addition, the resulting representations are easy to interpret, which also holds for certain aggregation steps (e.g., mean of all items in a cluster). Finally, building such representations is computationally efficient. The inference on powerful language models with millions of parameters for creating the embeddings needs considerable processing time.

Instead of using individual words as tokens for representing documents, one can also use longer sequences of words to increase the specificity of the representation, which are usually called n-grams. For instance, extracting not only frequent words but also frequent bi-grams from news reports would help to identify a larger set of entities such as frequently mentioned persons (combination of first and last name). However, the handling of longer sequences requires smart algorithmic strategies since the number of possible n-grams grows super-exponentially with respect to n.

Document Clustering Algorithms

Aggarwal and Zhai [2012] and Allahyari et al. [2017] discuss numerous text clustering algorithms that have been proposed. For the visual analysis of document collections, Non-Negative Matrix Factorization (NMF) (Xu et al. [2003]; Lee and

Seung [1999]), Latent Dirichlet Allocation (LDA) (Blei et al. [2003]), and k-Means (Lloyd [1982]) belong to the most frequently used clustering algorithms.

The idea of NMF is to decompose the positive matrix $\mathbf{X}$ that contains the TF-IDF-based document representations as columns into two positive matrices $\mathbf{U}$ and $\mathbf{V}$, where the number of columns of both matrices correspond to the cluster size. That is, the following objective function should be minimized (constraining $\mathbf{U}, \mathbf{V}$ to be positive):

$$J = \frac{1}{2} \|\mathbf{X} - \mathbf{U}\mathbf{V}^T\| \tag{2.4}$$

Each column of $\mathbf{U}$ can then be interpreted as the importance rating of the terms for the respective cluster, and each row of $\mathbf{V}$ as the cluster distribution of the respective document.

LDA is a probabilistic topic modeling method. The idea is that a fixed number of *latent topics* can describe a set of documents, and the goal is to infer these topics based on the data. Each document is said to be a sparse mixture of the k topics, and each topic is characterized by a distribution of words such that the topic distribution of a document can explain the words it contains (and vice versa). For each topic, a multinomial word distribution of the vocabulary is sampled from a Dirichlet distribution. LDA then assumes a generative process of each document. First, the multinomial topic distribution of each document is sampled from a Dirichlet distribution. Then, for each position in the document, a corresponding topic is sampled from the topic distribution and then the word from the corresponding multinomial distribution of the topic. The optimization process aims to find the parameters of said distributions that best explain the data.

One advantage of clustering documents with NMF instead of LDA is that the term frequencies can (and should) be weighted (e.g., with the inverse document frequency), which facilitates the incorporation of knowledge from external, larger corpora that are not part of the actual input. NMF is also often used in interactive topic modeling approaches by extending the set of optimization constraints based on user input to guide the clustering process (Choo et al. [2013]; Kim et al. [2017]).

This thesis, however, focuses on Spherical k-Means (Dhillon and Modha [2001]) that is based on k-Means. It belongs to one of the fastest clustering algorithms and typically leads to competitive clustering results. For instance, it performs better on a number of benchmark document collections compared to LDA and NMF (Lelu and Cadot [2021]).

Spherical k-Means

The original Euclidean k-Means algorithm (Lloyd [1982]) aims to find in an iterative way a partition $\mathbf{S}$ of the data set that minimizes its clustering objective. That is, every data item $\mathbf{x}_i$ gets associated with one of the k clusters such that the sum of the squared differences between data items and their assigned centroids $\boldsymbol{\mu}_j$ (i.e., mean of all associated items in the cluster) is minimal:

$$\underset{\mathbf{S}}{\operatorname{argmin}} \sum_{j=1}^{k} \sum_{\boldsymbol{x}_i \in S_j} \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\|_2^2 \tag{2.5}$$

The algorithm initially assigns every item to a cluster (e.g., with the k-Means++ strategy developed by Arthur and Vassilvitskii [2007]). Afterward, it optimizes the objective iteratively with a loop that comprises two steps. First, the cluster centroids $\boldsymbol{\mu}_j$ are recalculated based on the current assignments. Then, the assignments are updated, that is, for every data item $\boldsymbol{x}_i$ we find the cluster j such that the squared distance $\|\boldsymbol{x}_i - \boldsymbol{\mu}_j\|_2^2$ is minimized. The loop stops if the assignments do not change anymore or if a specific criterion is met, for instance, the maximum difference between subsequent cluster centroids is sufficiently small.

For document clustering use cases, Spherical k-Means (Dhillon and Modha [2001]) was proposed that applies the cosine similarity instead of the Euclidean distance to improve the clustering results. Every input vector is supposed to have unit length, and the centroids are calculated as the sum of the associated items, divided by the length to obtain unit vectors as well. The data items are then assigned to the cluster with the highest cosine similarity (which is equivalent to the dot product of the unit-length vectors). Let p, k be the number of non-zero vector entries of vectors $\mathbf{a}$ and $\mathbf{b}$, respectively, then the complexity of calculating the dot product $\mathbf{a}^\top \mathbf{b}$ is in $\mathcal{O}(\min(p, k))$, given a map-like data structure of the vectors. Thus, the sparser the input data, the faster the clustering.

Online Clustering of Text Streams

While several techniques have been developed that incorporate temporal aspects (Mei and Zhai [2005]; Peng et al. [2018]; Wang and McCallum [2006]; Zhang et al. [2010]; Stilo and Velardi [2016]; Gao et al. [2011]), only few support the online clustering of streaming data. Gil-García and Pons-Porrata [2010] proposed a graph-based dynamic hierarchical clustering algorithm, but it does not scale well with the number of documents since each incoming item has to be compared with all existing items. EvoBRT (Liu et al. [2015a]; Wang et al. [2013]) is an evolutionary multi-branch tree clustering algorithm based on Bayesian Rose Trees (Blundell et al. [2010]) and Bayesian Hierarchical Clustering (Heller and Ghahramani [2005]), but is also not efficient enough to handle large data sets in real-time. MStreamOne (Yin et al. [2018]) and EStream (Rakib et al. [2021]) are optimized for the online clustering of short text streams. They immediately assign each incoming item to a cluster in only one pass to improve the efficiency on streaming data.

Several online versions of k-Means have been proposed that approximate the k-Means objective with different strategies: process each element only once and update the cluster centroids greedily after each element (Zhong [2005]; Chakrabarti et al. [2006]), perform an approximated version of k-Means locally on batches and use these centroids as input for the global clustering (Ailon et al. [2009]), or perform clustering only on a cleverly chosen sample (Ackermann et al. [2010]; Braverman et al. [2011]). These online versions are fast, but they approximate the original k-Means objective and the number of clusters is fixed. Furthermore, only the first strategy of updating the centroids greedily leads to coherent clusters over time.

2.1.3 Dimensionality Reduction

Visualizing highly unstructured and multidimensional data such as text proves challenging due to the limits of human perception. Hence, many visual analytics approaches first apply dimensionality reduction (DR) methods to visualize data sets as two-dimensional plots. PCA (Tipping and Bishop [1999]), MDS (Cox and Cox [2008]), t-SNE (Van Der Maaten and Hinton [2008]), and UMAP (McInnes et al. [2018]) belong to the most commonly used DR algorithms in VA approaches.

Principal component analysis (PCA) is a linear transformation that projects a d-dimensional data set onto $k \leq d$ orthonormal axes that are given by the first k eigenvectors in descending order of the corresponding eigenvalues (Tipping and Bishop [1999]). Multidimensional scaling (MDS) tries to find a projection such that the distances according to a specific scaling (e.g., Euclidean metric) between points in the reduced space are preserved as best as possible (Cox and Cox [2008]).

Van Der Maaten and Hinton [2008] introduced t-SNE (t-Distributed Stochastic Neighbor Embedding) to better preserve local neighborhoods. The distances between points are converted into probabilities that capture the likelihood of them being neighbors, that is, the closer the points are, the higher the probability that they are neighbors. The goal of the algorithm is then to find a projection such that these probabilities are preserved as best as possible. In contrast to linear methods, this has the advantage that the distances between distant points do not matter as much as between close points. Points that are farther away in the original space have a similarly low probability of being neighbors, even if the absolute distances differ greatly.

UMAP (McInnes et al. [2018]) builds a weighted graph from the data points based on the k nearest neighbors of each point, with exponentially decaying weights based on the distance between points. It tries to find a projection that minimizes the cross-entropy loss between corresponding edges in the two graphs representing the low-dimensional and high-dimensional space, respectively.

2.1.4 Neural Networks

An artificial neuron is a function that takes a weighted sum of its inputs plus a constant value (the bias) and then transforms this value with a nonlinear activation function σ such as $\tanh x$ (Dreyfus [2005]). A *neural network* is a network of said nonlinear neurons, representing a more complex function. Given a training data set with pairs of input values and their corresponding output values, the goal in such a supervised setting is to learn the parameters (weights) of the neural network such that it maps the inputs in the training set to their corresponding outputs, but also computes correct or reasonable output values for new inputs which are not part of the training set (*generalization*).

Several neural network architectures have been developed. Hochreiter and Schmidhuber [1997] introduced long short-term memory (LSTM) recurrent neural networks that operate on data sequences by using the output of the last time step again as additional input for the next time step, which is particularly relevant for processing text. Multiplicative gate units improve the flow of information, leading to faster training times and better modeling of longer-range dependencies compared to previous recurrent architectures.

Vaswani et al. [2017] proposed the Transformer architecture that does not use recurrence and instead relies solely on attention and positional encoding to process sequences of tokens with blocks of encoders and decoders, which lead to new state-of-the-art results for several natural language processing tasks such as machine translation. The encoders process the input sequence and the decoders then predict token-by-token an output sequence based on the encoded input and the already predicted tokens. Several recent large language models like GPT-2 (Radford et al. [2020]) are based on variations of the Transformer architecture.

However, this thesis mainly deals with the traditional, feedforward neural network architecture comprising two or more fully-connected layers. Each layer

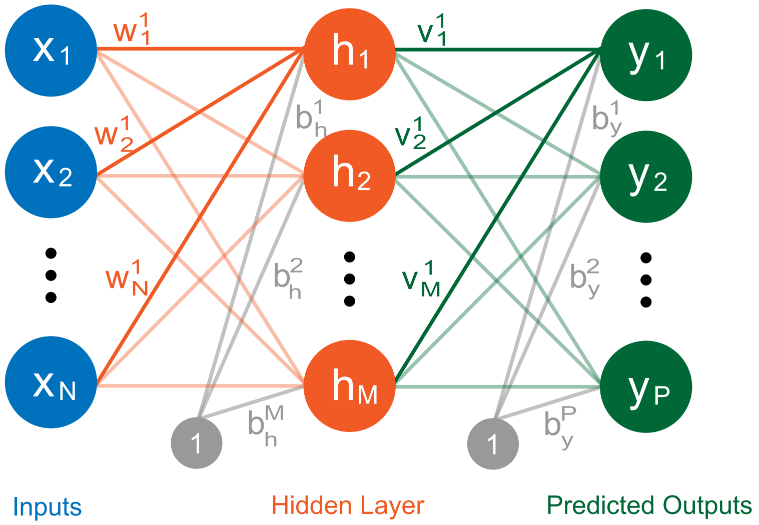


Figure 2.1 — Architecture of a simple feedforward neural network with one hidden layer, inputs $\boldsymbol{x}$, input-to-node weights $\boldsymbol{w^i}$, hidden node biases ${b_h}^i$, hidden node outputs h_i, node-to-output weights v_i^k, output biases b_y^k, and predicted outputs y_k.

contains a fixed number of neurons with a specific activation function, and each neuron in a layer takes the outputs of all neurons in the previous layer as input (or the actual input data if it is the first layer).

Figure 2.1 depicts a feedforward network architecture with one hidden layer. To compute the (scalar) output h_i of a hidden node i, the dot product of the inputs $\boldsymbol{x}$ with the trained weights of the node $\boldsymbol{w^i}$ plus a constant offset ${b_h}^i$ (the bias) is fed into a nonlinear, monotonically increasing activation function $\sigma(z)$:

$$h_i(\boldsymbol{x}) = \sigma(\boldsymbol{w^i} \cdot \boldsymbol{x} + {b_h}^i) \tag{2.6}$$

For regression tasks, the final outputs $y_k(\boldsymbol{x})$ are then computed from the outputs of the hidden nodes in the last layer as weighted sum (plus biases):

$$y_k(\boldsymbol{x}) = \sum_i h_i(\boldsymbol{x}) v_i^k + b_y^k \tag{2.7}$$

However, the last layer is often handled in a special way. For instance, if the goal is to train a classifier, the last layer is usually normalized with the Softmax function such that the output values always sum to one.

Neural networks are a powerful yet universal machine learning technique. With the sigmoid activation function, it has been shown that the feedforward network

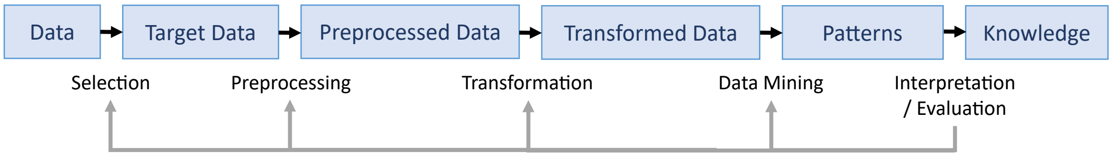


Figure 2.2 — The iterative *knowledge discovery in databases* process according to Fayyad et al. [1996].

is nonlinear regarding its parameters, which makes it more parsimonious than a linear or polynomial model (Dreyfus [2005]). This means that the number of parameters needed to approximate a certain function with a specific accuracy increases only linearly with the input dimension and not exponentially, as is the case with linear or polynomial models.

Neural networks can be applied in various settings (Bishop [1994]). They can learn to classify inputs, to predict certain quantitative values, or to model a possibly non-trivial function. The concrete training objective is specified with a loss function, that is, the training should learn weights that minimize this loss function. For instance, the mean squared error loss can be used for regression and the cross-entropy loss for classification tasks. The loss function often contains an additional regularization term to improve the generalization of the network (e.g., the squared sum of all weights).

The weights and biases are then usually learned with backpropagation (Linnainmaa [1970]; Rumelhart et al. [1986]) and stochastic gradient descent (SGD). Several optimization routines that implement an adapted version of SGD have been proposed such as Adam (Kingma and Ba [2014]). Instead of updating the weights after each training sample, training is usually done in an accumulative way as groups of mini-batches to improve the efficiency with parallel execution (Li et al. [2014]).

2.2 Visual Analytics

In 1989, Piatetsky-Shapiro chaired a workshop on *Knowledge Discovery in Real Databases* as there was "both a need and an opportunity for extracting knowledge from databases" due to the growing amounts of collected data (Piatetsky-Shapiro [1990]). Back then, this extraction process mostly refered to the challenge of applying machine learning techniques with the goal to replace or even surpass human experts. In other words, machine learning models should extract and encapsulate knowledge from large databases so that this knowledge can be repeatedly and automatically applied in the future.

Later on, the notion of *knowledge discovery in databases* (KDD) became more human-centric. In 1996, Fayyad et al. [1996] described it as a process "of mapping low-level data into other forms that might be more compact, more abstract, or more useful", involving several steps from data selection, data preprocessing, data transformation, data mining, and then interpretation of the resulting patterns to gain knowledge. Figure 2.2 outlines the stages and steps. Hence, the discovery process should ultimately lead to *human* insights rather than machine-centric models that are often difficult to understand. In this framework, the user has a central role as they have to take decisions at each step, including the possibility of going back to a previous step at any time to change decisions. The gained knowledge may also influence and feed into future iterations of this pipeline.

One of the core steps in the model refers to *data mining* that is about extracting patterns, which could involve "fitting a model to data; finding structure from data; or, in general, making any high-level description of a set of data" (Fayyad et al. [1996]). In 1999, Wong [1999] introduced the concept of *visual data mining*. Instead of applying statistics or more advanced algorithms to extract patterns automatically, the idea was to fuse these purely analytical processes with visualization:

> "[Visual data mining] integrates the human mind's exploration abilities with the enormous processing power of computers to form a powerful knowledge discovery environment that capitalizes on the best of both worlds."

Gershon et al. [1998] point out the advantages of using information visualization for recognizing patterns:

> "Visualization is the process of transforming data, information, and knowledge into visual form making use of humans' natural visual capabilities. With effective visual interfaces we can interact with large volumes of data rapidly and effectively to discover hidden characteristics, patterns, and trends."

The data mining step in the KDD model may also involve visualization as a means to communicate the extracted patterns, but visual data mining aims to take advantage of the visual channel for the actual mining process. That is, the visualization should actively enable users to find patterns themselves without relying completely on automated methods. Wong further argues that such systems should have an intuitive user interface with outputs that are easy to interpret.

After the September 11 terrorist attacks in the US, Thomas and Cook [2005] presented in 2005 an agenda to better prevent such attacks in the future by analyzing massive amounts of data with *visual analytics*:

> "Visual analytics is the science of analytical reasoning facilitated by interactive visual interfaces. People use visual analytics tools and techniques to synthesize information and derive insight from massive, dynamic, ambiguous, and often conflicting data; detect the expected and discover the unexpected; provide timely, defensible, and understandable assessments; and communicate assessment effectively for action."

Similar to visual data mining, visual analytics (VA) also aims to combine visualization techniques with data analytics, but there is a strong focus on *interactive* interfaces, on analyzing *large amounts* of data, and on finding *unexpected* patterns. The combination of cognitive power with automated analyses promises to tackle highly complex problems that human beings or machines would not achieve on their own. However, it also means that at least one person has to interact with the system and analyze the problem at hand over a possibly longer time span, which can be expensive and time-consuming. Thus, visual analytics fits best if the problem is only vaguely defined, if the task is too difficult to solve with automated methods, if it leads to a better performance (time- or quality-wise), or if it is vital to have a high trust in the findings (Keim et al. [2010b]).

The field of visual analytics was initially strongly connected with counter-terrorism, intelligence work, and the monitoring and assessment of ongoing situations, but its scope quickly broadened. Keim et al. [2010a] later characterized the field as follows:

> "Visual analytics combines automated analysis techniques with interactive visualisations for an effective understanding, reasoning and decision making on the basis of very large and complex datasets."

They characterized it as a process with different stages and transitions as depicted in Figure 2.3. To some extent, this process resembles the KDD pipeline, enriched with visual data mining, but there are some important differences.

First, the process explicitly models different paths to gain knowledge from data:

1. The potentially transformed data is visually mapped to an interface that the analyst can interact with to gain knowledge from the resulting visual

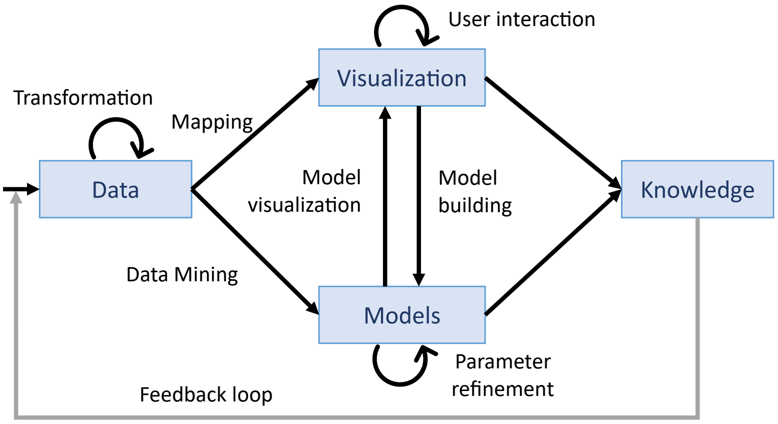


Figure 2.3 — The visual analytics process according to Keim et al. [2010a].

representations, which corresponds to the visualization reference model by Card et al. [1999]

2. Interactive model building is performed with advanced data mining methods that are applied to the potentially transformed data, and this model then may encapsulate knowledge. This path resembles the traditional, machine-centric definition of knowledge discovery in databases as characterized by Piatetsky-Shapiro [1990]

3. While interactive model building is performed with advanced data mining methods that are applied to the potentially transformed data, this model then serves as the basis for interactive visualizations from which the analyst can gain knowledge. These steps closely map to the KDD pipeline as defined by Fayyad et al. [1996] in combination with visual data mining (Wong [1999]).

4. An intertwined combination of the aforementioned paths.

Second, the process emphasizes that interactive visualizations help not only to explore the data but also to steer the underlying data mining model, leading to more refined automated data analyses.

Finally, it can be noted that the higher level of interactivity also constrains the space of feasible algorithms. They need to be efficient enough such that analysts do not need to wait unreasonably long for an update to complete after changing parameters.

Hence, visual analytics aims to fuse interactive visualization, advanced knowledge discovery, and visual data mining into a holistic, human-centric system for the analysis of complex and large data sets. While visual analytics can be applied in many different fields and settings, this thesis focuses on the analysis of document collections and multivariate data sets. The next two sections therefore introduce related VA approaches in these two fields.

2.3 Visual Analysis of Document Collections

The analysis of vast document collections is important for many stakeholders across different domains. News articles are a valuable source for historians to research events but also opinion changes in certain demographics (Allen and Sieczkiewicz [2010]). For investigating leaks, journalists need to harvest many documents in a short amount of time (Yimam et al. [2016]). It is also of vital interest for many business analysts, journalists, and stock traders to closely monitor social media and news platforms for important developments that inform or affect their decisions (Hu et al. [2012]; Oliveira et al. [2017]; Sul et al. [2017]. Brands monitor their exposure on social media (Hoffman and Fodor [2010]; Liu and Lopez [2016]) and analysts can infer business intelligence from news reports (Chung [2014]). In disaster management, first responders benefit from timely witness reports on social media (Beigi et al. [2016]; Thom et al. [2015]).

Visual analytics is well suited for the problem of analyzing social media posts and news reports. The data sets are often huge and the goals typically ill-defined. Many analysis tasks have an explorative nature, for example, if analysts want to monitor the most important themes people talk about on social media so that they can start a deeper investigative analysis whenever an important topic emerges. Such explorative tasks are difficult to solve with completely automatic methods as it is difficult to formulate the goal in a machine-readable way. In the case of the monitoring task example, how could one mathematically describe the concept of an important topic? It could be content-based, but it could also be related to an unexpected development in the context of certain events. Furthermore, it is still difficult for machines to fully understand text as this would require human-level intelligence.

This section therefore discusses related approaches that deal with the visual analysis of textual data, and in particular, the analysis of text collections. Kucher and Kerren [2015] provide a more general overview of text visualization techniques.

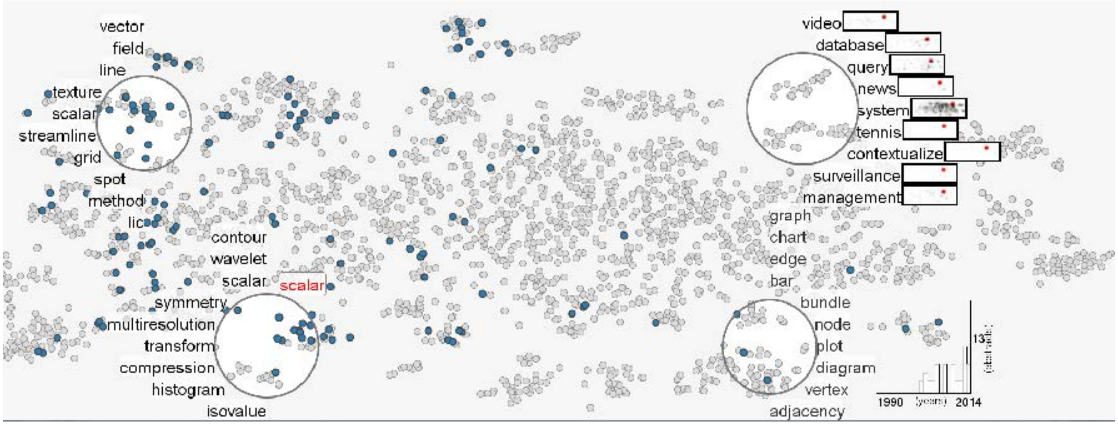


Figure 2.4 — Many approaches project documents onto two-dimensional maps. This figure is taken from DocuCompass (Heimerl et al. [2017]), which computes a t-SNE dimensionality reduction of documents and provides a lens that visualizes several features based on the target region.

2.3.1 Projection-Based Approaches

Several approaches project individual documents or textual tags onto two-dimensional plots (spatialization) such that analysts can explore related items and discover interesting patterns.

Document Projections

Wise et al. [1995] introduced the concept of transforming documents to a spatial, two-dimensional representation, enabling a more natural way of perceiving thematic patterns and relationships among documents. The Galaxies visualization projects high-dimensional representations of documents to points in a 2D scatter plot. Several approaches adopted this visual encoding to visualize text collections (Lagus et al. [1996]; Weippl [2001]; Paulovich and Minghim [2006]; Paulovich et al. [2008]; Chen et al. [2009]; Alsakran et al. [2011]; Paulovich et al. [2012]; Gansner et al. [2013]; Heimerl et al. [2017]). Alencar et al. [2012] review approaches for visualizing text documents or collections in greater detail.

These spatializations can be extended, made interactive, and combined with existing visualization techniques in several ways. Paulovich et al. [2012] enriched document projections with localized word clouds. Analysts can draw a polygon to interactively select documents in the projection, and the polygon will then be replaced by a word cloud containing the most relevant tags from this selection. Similarly, DocuCompass (Heimerl et al. [2017]) performs a t-SNE dimensionality reduction (see Section 2.1.3) and provides a lens that visualizes several features based on the documents under the lens, including related tags. Figure 2.4 depicts the approach. TwitterScope (Gansner et al. [2013]) projects tweets related to a keyword onto a dynamic map with MDS. STREAMIT (Alsakran et al. [2011]) shows documents as small particles in a force-directed simulation, changing dynamically as new documents are streamed. In StreamExplorer (Wu et al. [2018]), tweets belonging to an event are projected with self-organizing maps (SOMs).

These approaches facilitate an interactive exploration of similar documents, without requiring explicit search queries and rankings. However, they typically do not scale well to big data sets. Projecting many individual documents as dots on a map can lead to visual clutter, and the content typically only becomes apparent upon interacting with the map.

Tag Clouds

Instead of projecting individual documents, several approaches first extract relevant keywords from the collection to plot these extracted terms as a compact content summary onto a tag map or tag cloud. The concept of tag clouds dates back to 1976, but beginning with the late 1990s it started to become popular on the web as a way to present frequent search terms or user-generated tags (Viégas and Wattenberg [2008]). Initially, tag clouds were often regarded as a tool for designers and there was a strong focus on aesthetics and visual appearance (Seifert et al. [2008]; Viégas et al. [2009]). Bateman et al. [2008] found out that font size, font weight and color have a strong influence on which tags users typically select.

Later on, visualization research investigated the use of tag clouds for analytical tasks. Schrammel et al. [2009] conducted a task-driven study which showed that while semantically structured tag cloud layouts worked better than random layouts, a simple alphabetical list still performed best for finding specific tags. There was no difference regarding the search time for finding tags related to a topic or the ability to recall tags. Lohmann et al. [2009] concluded that the best way to arrange tags depends on the task at hand, which was corroborated by Felix et al. [2018]. Research from Sinclair and Cardew-Hall [2008] suggests that tag clouds have strengths for non-specific information discovery, and Wang et al. [2014] found out that semantically clustered word clouds can improve the understanding of large document collections. Word clouds typically only use single words (unigrams), however, descriptive tags to summarize documents for analytical tasks should include phrases with more than one word (Chuang et al. [2012a]).

Liu et al. [2015b] proposed a new technique for word cloud navigation which changes the word sizes dynamically while considering the mental map of users. Dörk et al. [2008] introduced VisGets to explore news items with linked visualizations. Dynamic search queries can be defined using an alphabetically sorted word cloud, a geographic map, and a date/time slider. Hovering over a tag highlights related tags and visual elements. Heimerl et al. [2014] presented a visual text analytics tool that heavily relied on word clouds with interactive features for filtering, co-occurrence highlighting and statistical insights.

Several efforts were made to generate more context-aware layouts. In most approaches, terms occurring in the same sentence are considered to be related. Adä et al. [2010] and Cui et al. [2010] applied multidimensional scaling to create context-preserving word clouds. Wu et al. [2011] improved upon this to generate layouts more reliably. Talin et al. [2010] clustered user-generated tags semantically based on their target resource. ReCloud (Wang et al. [2014]) applies a force-directed graph layout to produce semantically clustered word clouds on restaurant reviews. Xu et al. [2016] first constructed a similarity graph using word embeddings, transformed it then with multidimensional scaling and finally used force-directed methods to obtain dense layouts. TagSpheres (Jänicke and Scheuermann [2017]) aims to visualize hierarchical relations by arranging tags in circular bands, while at the same time placing related tags nearby. Endert et al. [2013] generated a 2D spatialization of the entire English Wikipedia with about four million documents. Several thousand important terms, phrases and snippets are extracted and placed in such a way that semantically similar tags are placed nearby. Analysts can zoom in until they finally retrieve individual Wikipedia articles.

TexTonic (Paul et al. [2019]) generates a hierarchical, clustered spatialization of millions of Wikipedia articles. At first, a general overview with clusters containing the main tags is presented. Users can subsequently zoom into clusters to reveal more terms and more fine-grained clusters. A search box is provided in case users do not find clusters they are interested in at first glance.

To visually compare document collections based on the time or location, Collins et al. [2009] proposed Parallel Tag Clouds in which each category (date, location, etc.) is represented by a column containing a list of extracted relevant terms. Analysts can hover over tags to highlight and follow its trajectory. The authors proved its scalability by applying it to 600,000 court documents. Figure 2.5 shows SparkClouds (Lee et al. [2010]) in action, which enriches tag clouds with line charts under each tag to show its popularity over time. Chi et al. [2015] applied rigid body dynamics to smoothly morph word clouds over time. Cui et al. [2010] combined their context-preserving word clouds with a significance line chart and generated multiple clouds for a selection of time steps

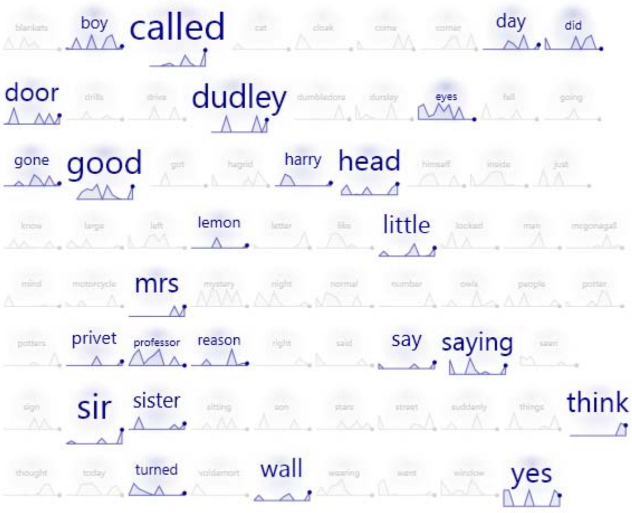


Figure 2.5 — Tag clouds are a popular way to visually summarize collections. This example is taken from Lee et al. [2010], which depicts the temporal evolution of individual tags with small line charts.

that were deemed most significant. Binucci et al. [2016] proposed animated word clouds to show the temporal evolution of real-time streaming data while trying to preserve the mental map of the user. ConcentriCloud (Lohmann et al. [2015]) merges word clouds from different documents and is optimized for the comparison of a few long documents such as books.

The introduction of context-aware and time-aware layouts have increased the utility of tag cloud-based approaches for visual analyses. However, research is still lacking with respect to non-animated layouts that are *both* more context-aware and also more time-aware. In addition, previous work often relies on extracting single keywords that can be difficult to interpret due to the limited context.

Trees and Patterns

Apart from tag cloud approaches, tree-based techniques were proposed to let analysts inspect structural patterns in large document collections. The work of Burch et al. [2013] combines similar tags that share a prefix to generate word cloud layouts that save space and quickly reveal different variants of terms. Wattenberg and Viégas [2008] presented an interactive word tree which renders the structure of sentences as an hierarchical tree, enabling analysts to explore how sentences continue in different variants. However, it is not meant to provide users with a general overview of the corpus as it requires a keyword

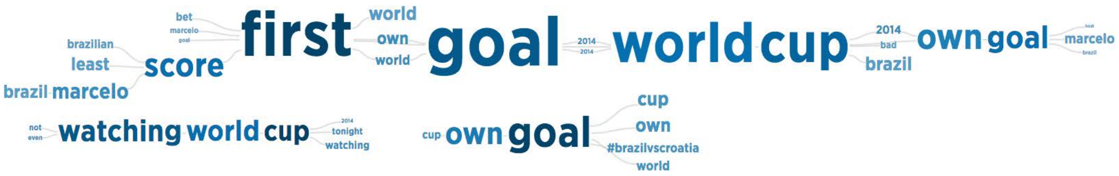


Figure 2.6 — Several approaches try to preserve linguistic structures so that it is easier to grasp concepts. This figure is taken from Hu et al. [2017] and shows part of a SentenTree visualization which aggregates tweets about the world cup.

search first to find an interesting starting point. Phrase Net (Van Ham et al. [2009]) generates node graphs from text to reveal syntactic or lexical relations based on a pattern query. Figure 2.6 shows part of a SentenTree visualization (Hu et al. [2017]), which extracts frequent sequential patterns from tweets and generates a node-link diagram trying to preserve the word order.

Jigsaw (Stasko et al. [2008]) is a tool for investigative journalists and law enforcement. It provides multiple coordinated views, but has a strong focus on detected entities and their connections.

2.3.2 Topic-Based Approaches

Early on, researchers facilitated clustering algorithms to visually structure and aggregate large text collections by grouping similar documents. However, the automatically extracted clusters may not correspond exactly to how humans would have defined the topics, the total number of topics is often small to fit into the visualization, and larger topics might overshadow smaller topics as well as outliers. The approaches can be broadly divided into two categories: some ignore the publishing date of documents, whereas others specifically incorporate it as part of the interactive visualization.

Non-Timestamped Data

Several approaches adopted LDA (see Section 2.1.2) to extract and visualize topics: Dredze et al. [2008] enriched emails with summative keywords, ParallelTopics (Dou et al. [2011]) uses rows of tag clouds to convey the content of topics, and Serendip (Alexander et al. [2015]) provides topic-based visual exploration of documents in a matrix-like visualization. HierarchicalTopics (Dou et al. [2013]) uses EvoBRT to cluster documents hierarchically. TopicPanorama (Wang et al. [2016]) lets analysts compare topics between different corpora with an interactive node-link diagram and is also based on EvoBRT. LDA and EvoBRT require significant processing time on larger data sets, though. Termite (Chuang

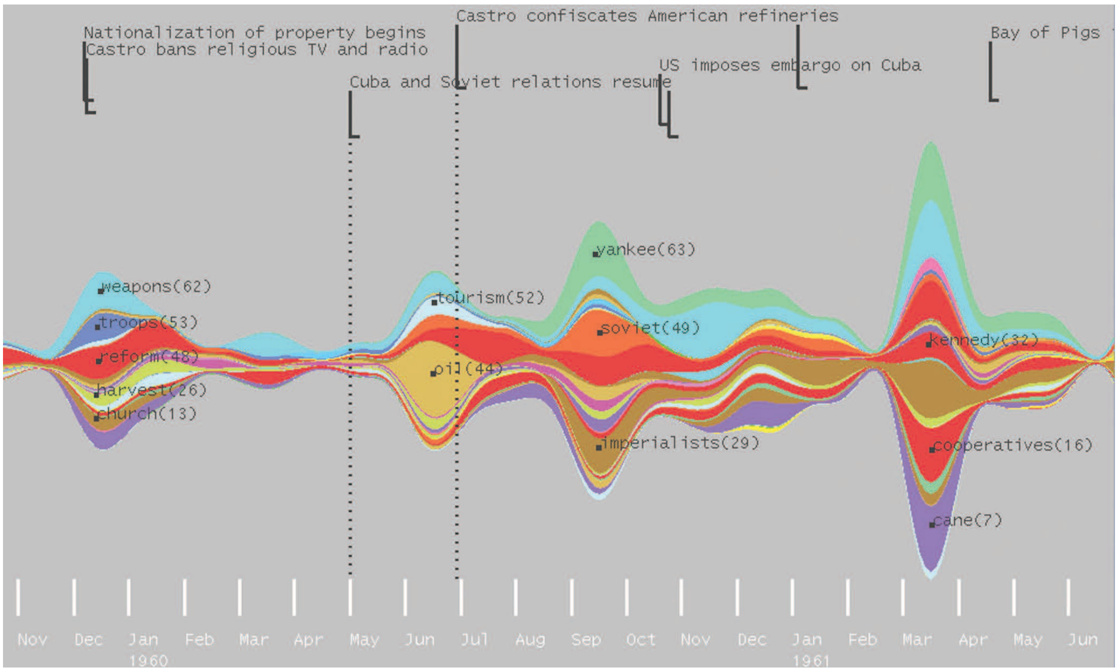


Figure 2.7 — The ThemeRiver visualization of topics over time (taken from Havre et al. [2002]) inspired many approaches for visually analyzing time-stamped documents.

et al. [2012b]) helps analysts to assess topic models in a tabular layout. They propose a seriation method that favors the natural reading order to quickly reveal relevant patterns. El-Assady et al. [2016] introduced topic-space views for visually analyzing conversations.

On the one hand, it can be challenging to interpret automatically derived topics (Chuang et al. [2012c]) and it is important for the analysis that clusters have suitable labels (Carpineto et al. [2009]). On the other hand, Alexander and Gleicher [2016] found out that while the quality of the topics seems to influence how easy it is for users to make sense of them, the visual representation has less of an effect.

As an alternative to completely automatic techniques, several works have investigated the use of interactive topic modeling for the analysis of document collections (Choo et al. [2013]; Hoque and Carenini [2019]; Park et al. [2018]; Yang et al. [2021]). Analysts are supposed to actively steer the clustering process for an improved visual representation and understanding, for instance, by merging or splitting topics.

Time-Dependent Documents

The publishing date of a document represents an important metadatum that can help to identify and shape topics. Some approaches process the complete data set once to extract topics solely based on the content and integrate the temporal metadata afterward in the visualization (Havre et al. [2002]; Hassan et al. [2014]; Heimerl et al. [2016]; Dörk et al. [2008]; Krstajić et al. [2011]; Liu et al. [2009]; Wang et al. [2012]). Others either utilize adapted clustering techniques that incorporate additional metadata such as the date into the clustering process itself (Cui et al. [2011, 2014]), or process the data set in bins, with each bin spanning a certain time range (e.g., one day), and then try to connect the resulting clusters between adjacent time steps afterward (Gad et al. [2015]; Krstajić et al. [2013]). However, all variants rely to a certain extent on a global view of the data set and do not easily support the online analysis of streaming data.

Several visual representations have been proposed to convey the temporal evolution of topics. ThemeRiver (Havre et al. [2002]) inspired many approaches to visualize the occurrence of topics over time in a streamgraph (Figure 2.7), resembling a river-like metaphor (Cui et al. [2011, 2014]; Heimerl et al. [2016]; Liu et al. [2009]; Sun et al. [2014]; Wang et al. [2012]). CloudLines (Krstajić et al. [2011]) visualizes the frequency of entities or events over time in rows. Each column in StoryTracker (Krstajić et al. [2013]) depicts clusters of news reports from the respective day, and visual connections between cells of neighboring columns reveal relationships between them. Similarly, columns composed of keywords in ThemeDelta (Gad et al. [2015]) represent specific date ranges and the brushing helps to trace the keywords over time.

2.3.3 Social Media Analysis

Shortly after the rise of new microblogging platforms such as Twitter, new approaches were developed to analyze social media posts. This section discusses relevant work that targets either offline analyses on static data sets that were collected in the past, or online analyses for the real-time analysis of currently posted items. See Wu et al. [2016] and Chen et al. [2017] for a more thorough analysis of social media visual analytics.

Offline Social Media Analysis

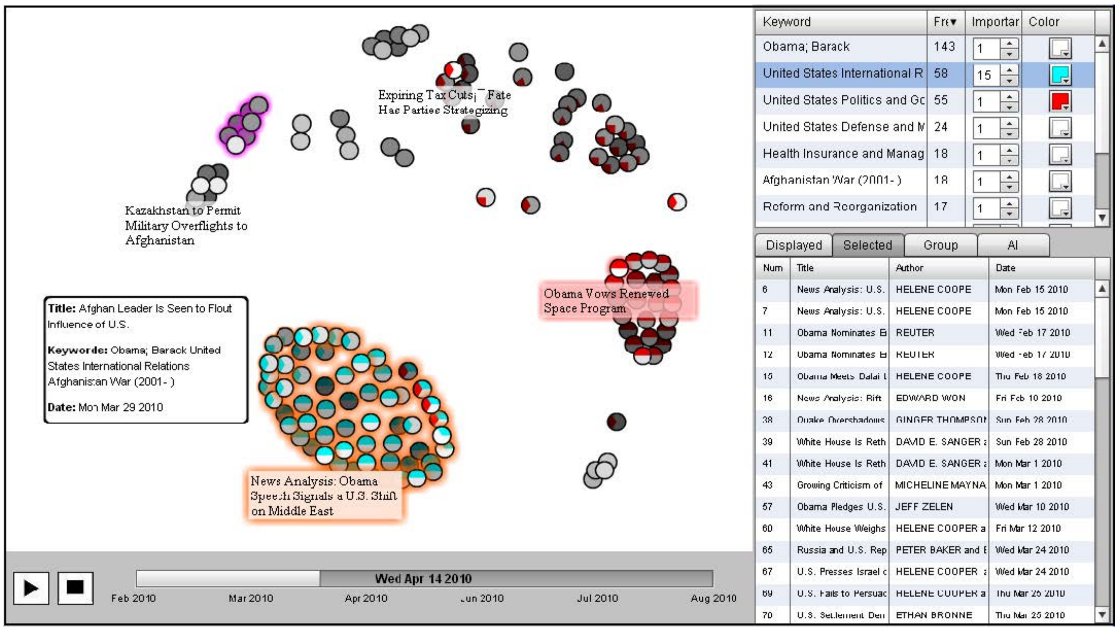
Vox Civitas (Diakopoulos et al. [2010]) relates social media posts to the respective video that was commented on and visualizes extracted keywords over time. I-SI (Wang et al. [2012]) is an architecture that extends ParallelTopics (Dou et al. [2011]) for analyzing social media data and latent topics using a high-performance computing cluster. ThemeCrowds (Archambault et al. [2011]) generates several tiles of multi-level tag clouds for each time span (e.g., days) to summarize twitter comments. LeadLine (Dou et al. [2012]) visualizes extracted topics in rows and integrates event detection and named entity recognition for visually analyzing text data. SentenTree (Hu et al. [2017]) was developed to summarize social media content while preserving the word order in a node-link graph (Figure 2.6). StanceVis Prime (Kucher et al. [2020]) classifies the sentiment and stance of social media posts from a specific topic and visualizes their temporal evolution. Other approaches (Viégas et al. [2013]; Wu et al. [2014]; Chen et al. [2021]) focus on visualizing network aspects of posts such as the information flow.

Harvesting shared geolocations of posts is a popular way to visually aggregate data. TwitInfo (Marcus et al. [2011]) lets analysts retrieve relevant tweets related to specified keywords and visualizes geolocalized tweets on a map. SensePlace2 (MacEachren et al. [2011]) visualizes tweet volumes with a geo-heatmap for situational awareness. Steiger et al. [2016] introduced the geographic, hierarchical self-organizing map to cluster geolocated tweets. TopoText (Zhang et al. [2018]) aggregates and visualizes spatial topics on a map across multiple scales.

Real-Time Social Media Analysis

Processing and visualizing streaming data is challenging in several ways. Data has to be processed with fast algorithms that support incremental updates (Rohrdantz et al. [2011]), and dynamic visualizations have to be developed that preserve the mental map of users (Krstajić and Keim [2013]).

Dörk et al. [2010] introduced one of the first visual analytic systems to follow tweets of an ongoing event, which includes a ThemeRiver-inspired visualization conveying the temporal evolution of important topics. Each stemmed word represents a topic, which limits the expressiveness of the topics, though. Twitcident (Abel et al. [2012]) automatically fetches relevant tweets for incidents that have been broadcast and provides a faceted search with enriched metadata, including named entity recognition. Liu et al. [2016] proposed a tree- and sedimentation-based visualization of topics in text streams that uses EvoBRT (Liu et al. [2015a]) for clustering. STREAMIT (Alsakran et al. [2011]) and TwitterScope (Gansner et al. [2013]) project items to a dynamic 2D plot. STREAMIT applies a physical model to ensure the continual evolvement while new documents are received, supporting hundreds of documents (Figure 2.8). TwitterScope (Gansner et al. [2013]) projects tweets related to a keyword onto a map with MDS, either using cosine- or LDA-based similarity, and aims to maintain the relative position



Title: Afghan Leader Is Seen to Flout Influence of U.S.

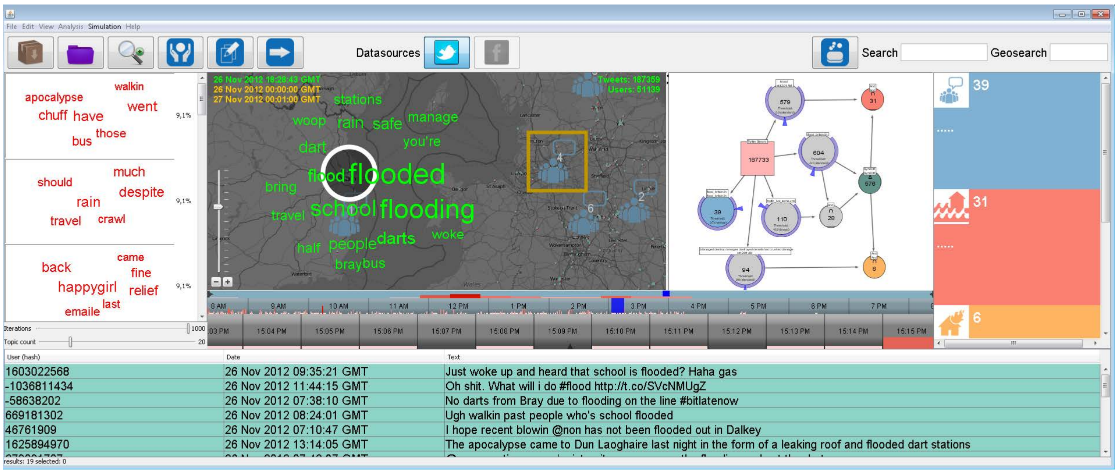
Keywords: Obama; Barack United States International Relations Afghanistan War (2001-)

Date: Mon Mar 29 2010

Keyword	Fre▾	Importar	Color
Obama; Barack	143	1	□
United States International R	58	15	🟦
United States Politics and Gc	55	1	🟥
United States Defense and M	24	1	□
Health Insurance and Manag	18	1	□
Afghanistan War (2001-)	18	1	□
Reform and Reorganization	17	1	□

Num	Title	Author	Date
6	News Analysis: U.S.	HELENE COOPE	Mon Feb 15 2010
7	News Analysis: U.S.	HELENE COOPE	Mon Feb 15 2010
11	Obama Nominates B	REUTER	Wed Feb 17 2010
12	Obama Nominates B	REUTER	Wed Feb 17 2010
15	Obama Meets Dalai L	HELENE COOPE	Thu Feb 18 2010
16	News Analysis: Rift	EDWARD WON	Fri Feb 19 2010
38	Quake Overshadows	GINGER THOMPSON	Sun Feb 28 2010
39	White House Is Reth	DAVID E. SANGER a	Sun Feb 28 2010
41	White House Is Reth	DAVID E. SANGER a	Mon Mar 1 2010
43	Growing Criticism of	MICHELINE MAYNA	Mon Mar 1 2010
57	Obama Pledges U.S.	JEFF ZELEN	Wed Mar 10 2010
60	White House Weighs	HELENE COOPER a	Fri Mar 12 2010
65	Russia and U.S. Rep	PETER BAKER and E	Wed Mar 24 2010
67	U.S. Presses Israel c	HELENE COOPER a	Wed Mar 24 2010
69	U.S. Fails to Persuac	HELENE COOPER a	Thu Mar 25 2010
70	U.S. Settlement Den	ETHAN BRONNE	Thu Mar 25 2010

Figure 2.8 — STREAMIT (taken from Alsakran et al. [2011]) continuously visualizes streamed documents as dots in a dynamic force-directed layout, but this does not scale well to hundreds of thousands of documents.



User (hash)	Date	Text
1603022568	26 Nov 2012 09:35:21 GMT	Just woke up and heard that school is flooded? Haha gas
-1036811434	26 Nov 2012 11:44:15 GMT	Oh shit. What will i do #flood http://t.co/SVcNMUgZ
-58638202	26 Nov 2012 07:38:10 GMT	No darts from Bray due to flooding on the line #bitlatenow
669181302	26 Nov 2012 08:24:01 GMT	Ugh walkin past people who's school flooded
46761909	26 Nov 2012 07:10:47 GMT	I hope recent blowin @non has not been flooded out in Dalkey
1625894970	26 Nov 2012 13:14:05 GMT	The apocalypse came to Dun Laoghaire last night in the form of a leaking roof and flooded dart stations

results: 19 selected: 0

Figure 2.9 — Several social media analysis approaches harvest the shared geolocation of posts to aggregate many items. This figure depicts the user interface of the ScatterBlogs2 system (taken from Bosch et al. [2013]), which overlays relevant tags onto a geo-map and also provides means to configure filters interactively.

of nodes on each update that happens every minute. Representing posts as dots has the advantage that all changes are visually apparent, but it does not scale well to hundreds of new posts each second due to the increased visual clutter.

Whisper (Cao et al. [2012]) visualizes the diffusion of information on Twitter regarding different topics in real-time, with updates every five minutes. The sunflower-like visualization in which tweets are represented as dots on a map integrates the geolocation of tweets. The comprehensive ScatterBlogs system (Thom et al. [2012]; Chae et al. [2012]; Bosch et al. [2013]) visualizes term usage anomalies from geolocated tweets and was later extended with an event detection algorithm, filter methods, and means to create and train classifiers interactively (Figure 2.9). Their case study (Thom et al. [2015]) shows that situation awareness domain experts consider the real-time analysis of social media content to be useful, e.g., for disaster assistance. However, the percentage of geolocated tweets has steadily decreased in recent years, which renders approaches that rely on geo-annotations less useful.

StreamExplorer (Wu et al. [2018]) made it possible to visually analyze non-geolocated social streams with tens of thousands of posts on a budget PC. It first detects important time periods (*events*), and tweets belonging to an event can then be clustered based on GPU-assisted self-organizing maps (SOMs). Analysts can apply several interactive lenses, for instance, the word cloud lens, to investigate areas of the map and refine the SOMs interactively.

However, existing approaches for the real-time visual analysis of social media posts either do not scale well to high-velocity streams or rely on additional meta-data for filtering. Some approaches exploit the geolocation of posts to operate on subsets of the data based on geographical regions, whereas others implement event detection algorithms to extract batches of related posts, which can lead to delays.

2.4 Visual Multivariate Analysis

When dealing with tabular data sets, a typical research question is whether variables have any kind of relationship to each other. For instance, one might have conducted a survey on voting intentions and now the goal is to investigate whether the age of the participants *correlates* with the expressed intention to vote for a certain party. Similar tasks arise when working with text data. The previous section discussed related approaches to the visual analysis of document collections, which mainly focus on conveying and exploring the *textual content*.

However, we might also be interested in exploring the relationships between, for example, tokens, extracted entities, or document attributes.

In statistics, the term *correlation* often relates to the Pearson correlation coefficient that measures the linear correlation between two variables (and is often interpreted together with tests for statistical significance). This thesis, however, uses the term *correlation* in a broader, more information theoretical sense to define any kind of relationship between variables that relates to an interesting pattern which questions the assumption of independence (even if it is only an artifact of the sample).

There are many ways to analyze the correlations between two variables (which could be the same variable but in two data sets), including bivariate statistics and straightforward visualizations. In the case of the voting example, one could visualize the relationship by plotting the variables *age* and *voting intention* for a specific party as a scatter plot. The analyst is interested in understanding how the age of participants may partially explain (or predict) the target outcome (vote on election day), so *age* would be categorized as independent variable and *voting intention* as dependent variable.

However, the analysis goal could become more complex and the analyst might now want to understand how age and profession *combined* correlates with the voting intention of the participants. *Multivariate analysis* deals with such correlations that involve more than one independent variables and one (or more) dependent variable (Raykov and Marcoulides [2008]). In theory, many methods can be extended so that they can be applied to more than two variables. In practice, though, multivariate analysis is much more challenging due to the curse of dimensionality and the limits of human perception.

The sparsity of data sets usually increases with the number of variables because, otherwise, the size of the set would increase exponentially. As a result, it becomes increasingly difficult to estimate the value of any given point in the multidimensional space. This can pose a serious problem for estimating the mutual information, for instance (Doquire and Verleysen [2012]).

The increasing complexity of data sets can also lead to ambiguity, analogous to the multi-solution problem in mathematics. With more variables, there are also more options for modeling or explaining the same target outcomes without additional constraints. This is why Amar and Stasko [2004] argue that correlation methods for determining multivariate relationships should be combined with user guidance, motivating the development of novel data visualization techniques for explaining multivariate data sets. This section therefore discusses related approaches for the visual analysis of multivariate data.

2.4.1 Multivariate Data Visualizations

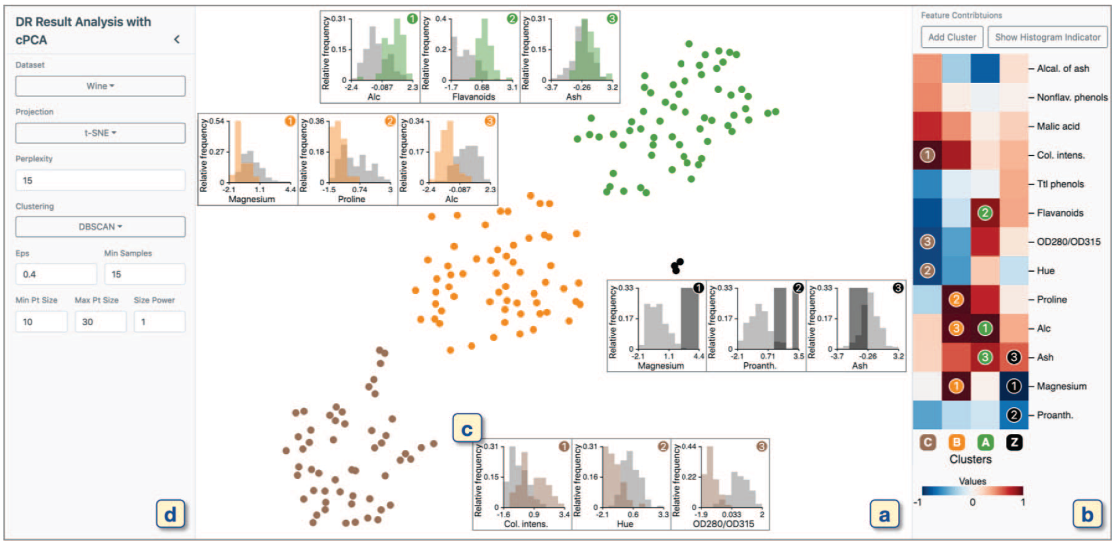
Scatter plot matrices (Carr et al. [1987]) and parallel coordinates (Inselberg [1985]) are widely used to visualize multivariate data sets, but they do not scale well to many variables (Barlowe et al. [2008]). Parallel coordinates (PCPs) have the advantage over scatter plot matrices that the required screen space only grows linearly with the number of variables, but they also suffer from several shortcomings. The order of the axes has an important influence on how relationships are perceived and between which variables, larger data sets easily lead to overplotting, and line-tracing becomes difficult with an increasing number of variables (Heinrich and Weiskopf [2013]).

There is ongoing research on how to improve parallel coordinates for large data sets. Artero et al. [2004] proposed frequency- and density-based PCPs by discretizing the axes into many bins. Johansson et al. [2005] cluster the data set to only plot a high-resolution texture for each cluster, whereas Janetzko et al. [2016] cluster the data sets such that they can overlay each axis with density plots based on the clusters. Novotný and Hauser [2006] apply a focus+context strategy that plots an aggregated plot based on the binned data and only draws the high-resolution data on demand. Richer et al. [2018] perform hierarchical aggregation to make the visual analysis scalable. Curve bundling (Heinrich et al. [2011]) was developed to help analysts perceive clusters in PCPs. Heinrich and Weiskopf [2013] provide a more thorough overview of proposed PCP techniques and open challenges.

2.4.2 DR-Based Approaches

Dimensionality reduction (DR) methods, including MDS (Cox and Cox [2008]), PCA (Tipping and Bishop [1999]), t-SNE (Van Der Maaten and Hinton [2008]), and UMAP (McInnes et al. [2018]) help to make multivariate data sets visually accessible. For instance, the set of independent variables could be reduced to two dimensions such that a scatter plot can visualize the relationship by mapping the dependent variable to the color of the dots. Section 2.1.3 introduces the common DR methods in greater detail. However, one major challenge of VA approaches that apply dimensionality reduction is that it is usually difficult to interpret findings from such reduced plots. Most DR methods assume that similar or nearby items in the original space should also be placed nearby in the reduced space, but for many methods it is not obvious how a visual pattern in the reduced space maps back to the original space. Furthermore, appearing patterns could also be misleading due to the inherent information loss of the process.

Thus, the crucial question for many analysts is how they can relate findings



DR Result Analysis with cPCA

Dataset: Wine

Projection: t-SNE

Perplexity: 15

Clustering: DBSCAN

Eps: 0.4

Min Samples: 15

Min Pt Size: 10

Max Pt Size: 30

Size Power: 1

Figure 2.10 — Dimensionality reduction methods are a popular way to visualize high-dimensional data sets, but for many use cases it is important to relate findings back to the original space. This is the user interface of the ccPCA system (taken from Fujiwara et al. [2020]). For each cluster in the reduced space, it visualizes which features and feature ranges distinguish the respective cluster from the others the most.

in the reduced space back to the original space (Sedlmair et al. [2012]). In the work of da Silva et al. [2015], the background color in the plot conveys the dominant dimension of each neighborhood, which is derived from an Euclidean- or variance-based ranking. Chatzimparmpas et al. [2020c] developed t-viSNE for visually exploring the quality of t-SNE projections and the impact of hyperparameters, as well as for understanding clusters and related dimensions of interest. Fujiwara et al. [2020] proposed ccPCA that clusters the result of the DR method and then visualizes for each cluster which features and feature ranges distinguish the respective cluster from the others the most. Figure 2.10 shows the user interface of the system. The cluster-specific ranking of the features and corresponding histograms support the analysis of data sets with many variables. However, since the clustering is done in the reduced space, the method may miss more complex relationships.

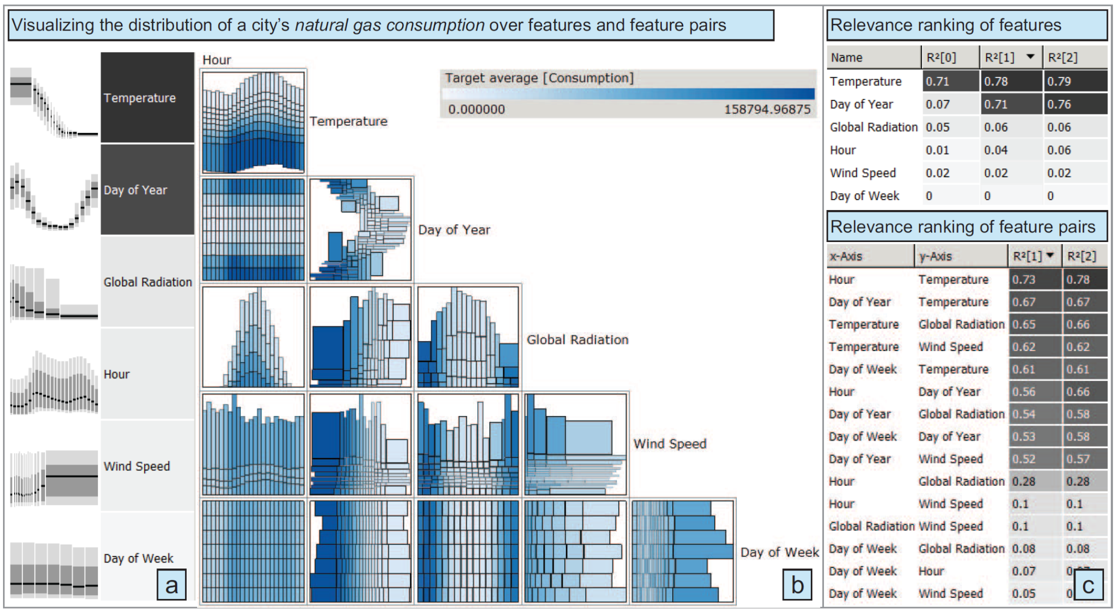
Another way of solving the mapping problem is to develop and apply more interpretable DR methods. The approach of Gleicher [2013] projects items onto user-defined and, thus, interpretable dimensions, for instance, as a linear combination of the selected variables. Garrison et al. [2021] aim to generate such interpretable dimensions automatically. They iteratively run several factor analyses of mixed data on the data set to identify bundles of dimensions that exhibit similar correlation coefficients, and each bundle is then only represented by its first or second principal component in the reduced representation. Embeddings based on generalized barycentric coordinates such as RadViz Deluxe (Cheng et al. [2017]) preserve the relation to the original space to some extent, but they work best if the data items have a dominant dimension.

2.4.3 Subspace-Based Approaches

Subspace extraction is a promising strategy to reduce the number of dimensions that have to be processed (e.g., for subsequent clustering methods) and visualized (Assent et al. [2007]; Alemzadeh et al. [2017]; Günnemann et al. [2010]; Hund et al. [2016]; Jackle et al. [2018]; Krause et al. [2017]; Müller et al. [2008]; Tatu et al. [2012a,b]; Wang and Mueller [2018]). Tatu et al. [2012a] employ an algorithm that detects interesting subspaces which are then grouped by similarity. Parallel coordinate and scatter plots display the data set in each subspace. SubVis (Hund et al. [2016]) uses the OpenSubspace framework (Müller et al. [2011]) to first extract subspaces and then find clusters within these subspaces. They apply multidimensional scaling to provide a visual overview of all clusters, from which analysts can select similar subspace clusters. The aggregation table visualizes the distributions of all related dimensions of a particular cluster. SeekAView (Krause et al. [2017]) supports building and refining subspaces interactively, including suggestions for interesting dimensions, for instance, based on how useful the respective variable is for predicting a target variable.

2.4.4 Iterative Approaches

Explorative and iterative approaches help to scale the analysis of large multivariate data sets, but they also help to let analysts derive sparse and justified models, which is important for clinicians, for instance (Dingen et al. [2019]). Voyager (Wongsuphasawat et al. [2016, 2017]) supports open-ended and focused exploration of multivariate data sets with automated recommendations and interactive chart specifications. DICON (Cao et al. [2011]) visualizes multidimensional clustering results for cluster interpretation and comparison. Scherer et al. [2011] introduced regressional feature vectors to enable visual sketch-based queries and to explore interesting scatter plots. Behrisch et al. [2015] propose a feedback-driven approach that iteratively learns the preferences of the user to make valuable suggestions for further explorations. Turkay et al. [2012] present a method to interactively generate representative factors that combine several data points across dimensions in order to reduce the number of dimensions. For geospatial data sets, specific methods were developed that facilitate maps (Goodwin et al. [2016]; Malik et al. [2012]).



Relevance ranking of features

Name	R²[0]	R²[1] ▼	R²[2]
Temperature	0.71	0.78	0.79
Day of Year	0.07	0.71	0.76
Global Radiation	0.05	0.06	0.06
Hour	0.01	0.04	0.06
Wind Speed	0.02	0.02	0.02
Day of Week	0	0	0

Relevance ranking of feature pairs

x-Axis	y-Axis	R²[1] ▼	R²[2]
Hour	Temperature	0.73	0.78
Day of Year	Temperature	0.67	0.67
Temperature	Global Radiation	0.65	0.66
Temperature	Wind Speed	0.62	0.62
Day of Week	Temperature	0.61	0.61
Hour	Day of Year	0.56	0.66
Day of Year	Global Radiation	0.54	0.58
Day of Week	Day of Year	0.53	0.58
Day of Year	Wind Speed	0.52	0.57
Hour	Global Radiation	0.28	0.28
Hour	Wind Speed	0.1	0.1
Global Radiation	Wind Speed	0.1	0.1
Day of Week	Global Radiation	0.08	0.08
Day of Week	Hour	0.07	0
Day of Week	Wind Speed	0.05	0

Figure 2.11 — Binning strategies help to make the visual analysis of large data sets scalable through aggregation. This figure depicts the approach of Mühlbacher and Piringer [2013] that allows to iteratively define and validate regression models in order to analyze multivariate relationships.

Several approaches rank variables and pairs of variables according to statistical correlation factors (Eichner et al. [2020]; Malik et al. [2012]; Piringer et al. [2008]) or classification metrics such as separability (Tatu et al. [2009]). Barlowe et al. [2008] visualize partial derivatives of the dependent variable with regard to the independent variables and analysts can iteratively explore multi-correlations. SmartStripes (May et al. [2011]) ranks and visualizes correlations based on *partitions* of the input features. Analysts can iteratively select multiple partitions to explore more complex relationships.

Zhang et al. [2015] developed the correlation map to visualize pairwise correlations between variables in a graph. The work of Klemm et al. [2016] visualizes correlations of up to three variables with regard to a target variable in a 3D cube.

2.4.5 Model-Building and Partitioning

Model building and partitioning play an important role in several approaches. INFUSE (Krause et al. [2014]) supports interactive model building by visualizing the predictiveness of features according to several feature selection algorithms. An integrated visual analytics approach to build logistic regression models was introduced by Zhang et al. [2016]. Dingen et al. [2019] developed RegressionExplorer to let analysts build and compare different regression models.

The approach of Mühlbacher and Piringer [2013] allows to iteratively define and validate regression models. They partition the input features into bins and visualize the target variable over binned features and pairs of features in a heatmap, ranked by their usefulness in predicting the target variable (Figure 2.11). Bernard et al. [2014] developed a system that operates on partitions of the input space as well, but with the goal to find multivariate relations between specific bins across attributes. They focus on detecting conditions that are statistically significant. Analysts can select one or several bins to reveal associated clusters with related bins based on pairwise mutual information.

Such binning (or partitioning) strategies make the visual analysis of large data sets scalable through aggregation. Another advantage is that even linear models can express certain non-linear relationships if the atomic unit is a bin (i.e., value range) and not a global variable anymore.

While uni- or bivariate metrics (e.g., Pearson correlation or mutual information) to rank features or cluster data items can offer statistically sound results, they bear the risk that analysts miss more complex correlations. For instance, there are cases in which only a combination of multiple variables may explain a specific behavior of the target variable, but looking at the individual variables or pairs of variables separately would not reveal a pattern.

2.4.6 Decision Trees and Neural Networks

Decision trees are commonly used to visualize different outcomes of the target variable depending on different splits of the input variables. BaobabView (Van Den Elzen and Van Wijk [2011]) allows analysts to interactively construct and refine such decision trees (Figure 2.12), and Liu et al. [2018] extended this concept for building and improving tree boosting models. Mühlbacher et al. [2018] focus on building pareto-optimal decision trees as a trade-off between accuracy, complexity, and interpretability. Neto and Paulovich [2021] convert the paths of a random decision tree forest into logic rules which are then visualized in a matrix.

Decision trees are a popular choice because they are usually easy to comprehend and interpret. However, continuous relations are difficult to model (e.g., the relationship between the horsepower and the acceleration time of a car) and slightly more complex patterns and 'splits' (e.g., two distinct ranges of a variable) can lead to overly complex trees, which makes it difficult for analysts to trace single paths.

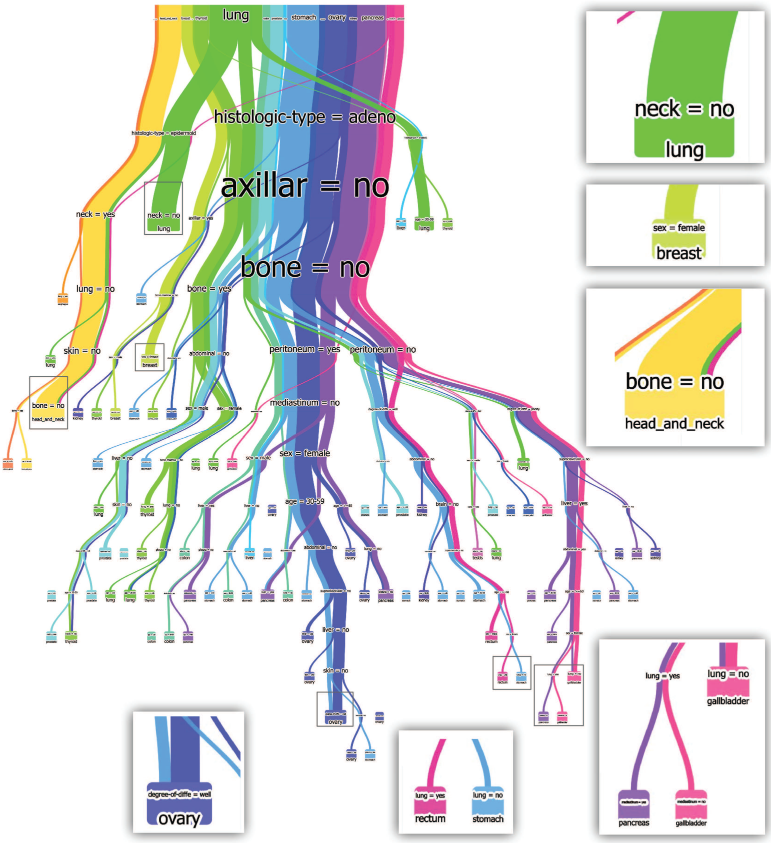


Figure 2.12 — Decision trees have the advantage that they are easier to interpret. Here, a decision tree is visualized with BaobabView that was trained on a primary tumor location data set (Figure taken from Van Den Elzen and Van Wijk [2011]).

Visualizing (deep) neural networks has become a popular research focus in recent years. In most cases, the approaches focus on explaining and debugging the *models*, for instance, visualizing which pixel regions are most supportive for the prediction (Zintgraf et al. [2017]), explaining predictions of convolutional neural networks with surrogate decision trees (Jia et al. [2020]), or visualizing activation patterns to understand deep learning models (Kahng et al. [2018]). Liu et al. [2017] provide an overview of visual analytic approaches for understanding, debugging and refining machine learning models, Choo and Liu [2018] for explainable deep learning, and Sacha et al. [2019] for assisting machine learning. Some approaches, though, indirectly also reveal structures of the data. CNNVis (Liu et al. [2017]) visualizes the learned features of neurons and their interactions to analyze image-based CNNs. While the aim is to refine and debug such networks, the resulting visualizations also offer a glimpse into the structure of the training data set, including prevalent features of the images and different clusters (e.g., cats and dogs). Likewise, LSTMVis (Strobelt et al. [2018]) allows analysts to retrieve similar sentences and paragraphs in the corpus, even though the approach is about visualizing hidden states of recurrent neural networks.

2.5 Artificial Intelligence for Visual Explainability

The interplay between model building and interactive visualization is one of the core characteristics of visual analytics. In most cases, the part in the process that is characterized as *model visualization* refers to visualizing or integrating the *outputs* of the model in the interfaces (e.g., predictions or classifications). For instance, we may train a document classifier which then steers the faceted search interface for browsing the documents based on their predicted categories. Another more recent example is training a machine learning model in order to use the prediction error as a proxy metric for interestingness or novelty, which Tkachev et al. [2021] proposed for spatiotemporal volume data, and Knittel et al. [2018] for highlighting text regions of interest in documents. This idea of applying ML techniques for advancing visualization approaches is also called *ML4VIS*. Wang et al. [2021] provide a survey on recent approaches.

Inspecting the model itself rather than what it outputs might also lead to interesting insights, though. If we fit a model to the data or train a classifier, this model has to capture some charateristics of the underlying data set to perform its task, it is supposed to *learn* relevant relationships in the data during training. *Transfer learning* (Pan and Yang [2010]) is one exemplary field that exploits this observation. One first trains a model on a large annotated data set, and then continues learning for a short amount of time on the actual data set. The target data set may not have enough labeled data items to train a bigger model from scratch, but with transfer learning we may still obtain a well-performing model, given that the data sets are sufficiently related. In addition, it has also been shown that neurons in deep neural networks may capture more complex and abstract concepts from the images it was trained on, without explicitly labeling these concepts during training (Bau et al. [2020])

Hence, the inner structure of the model encapsulates certain relationships of the data set, which may help to gain deeper insights into the data if we are able to visualize and understand it. One can conceptualize this observation and integrate it into the visual analytics pipeline to scale the analysis of large multivariate and textual data sets. The idea is to exploit **A**rtificial **I**ntelligence for visually e**X**plaining relationships and characteristics of the data set (AIX).

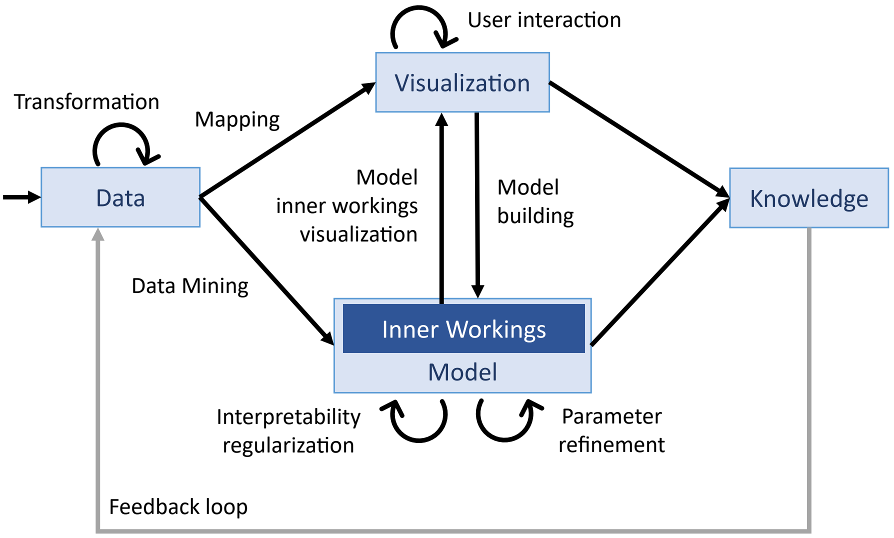


Figure 2.13 — The AIX process that exploits artificial intelligence for visually explaining data sets. Similar to the visual analytics process by Keim et al. [2010a], there are different paths to gain insights into data, for instance, by mapping the data to visual encodings. However, the goal of the interactive model building here is to primarily visualize the inner workings of the resulting model and not its outputs. Regularization techniques may be required to improve the interpretability of more complex models.

Figure 2.13 depicts the AIX process that is adapted from the visual analytics process by Keim et al. [2010a]. We fit a model to the data and then visualize the *inner workings* of the resulting model such that analysts can interactively inspect it with the goal of finding relevant patterns. That is, we perform the training not for the sake of predicting values but to visually explain the underlying data set. One major advantage is that this framework can already benefit from existing techniques to scale the training or fitting process since working with big data sets is one of the cornerstones of artificial intelligence, especially in the field of machine learning.

The concept also extends to non-predictive or unsupervised techniques such as clustering and dimensionality reduction. For instance, understanding why certain clusters were formed and how they differ to each other can lead to interesting insights regarding the composure and structure of the data set.

2.5.1 Regularizing Visual Interpretability

Unfortunately, it is often challenging to understand the inner workings of a model, particularly if it is rather complex and, thus, difficult to visualize. We therefore need techniques that are sufficiently complex, efficient to train or fit to, and *visually interpretable* at the same time. A simpler model is often easier to interpret but may fail to capture relevant non-linear relationships. Likewise, a powerful and interpretable but inefficient model is less suited for the analysis of large data sets within a reasonable time frame.

In machine learning, regularization is often used to increase the generalizability of models. In AIX, though, we need to think about regularizing the model for visual interpretability. That is, the model should ideally learn the characteristics of the data set in a way that analysts can interpret it, even if it may impact the prediction performance negatively. The level of regularization depends on the complexity of the model. A higher degree of freedom also leads to an increasing number of ways to encode the same relationships, with varying degrees of intelligibility. As a result, the AIX process (Figure 2.13) explicitly models the impact of this regularization due to the importance of having interpretable models.

For some models such as logistic regressions or support vector machines, traditional regularization techniques that are meant to improve the performance of a model on unseen data may also improve the understandability of the model since they are often a "proxy metric for comprehensibility" according to Gleicher [2016]. For more complex models such as neural networks, though, it is more challenging to increase the visual interpretability with weight penalties.

The strive for visually interpretable models is not new. Hastie and Tibshirani [1986] introduced Generalized Additive Models (GAMs) which are expressed as a sum of smooth functions, in which each function operates on exactly one input variable. Hence, even though GAMs can capture non-linearities of individual variables, the resulting models are still visually interpretable (Hastie and Tibshirani [1987]). Fayyad et al. [1996] point out that decision trees as data mining method have the advantage that humans can interpret them. Heimerl et al. [2012] proposed an approach to train document classifiers interactively and iteratively, which also incorporates a visualization of the current state of the trained support vector machine model. Mühlbacher et al. [2018] developed an approach to find pareto-optimal decision trees regarding several objectives such as prediction accuracy and interpretability. Lakkaraju et al. [2016] presented interpretable decision sets, a classification framework that consists of comprehensible, independent if-then rules. However, most approaches focus on building good *predictive* models based on data, less on building models that *explain* the data.

2.5.2 AIX vs. Predictive Model Building and XAI

While the concept of building interpretable predictive models is closely related to the concept of using artificial intelligence for explaining data sets (AIX), they differ regarding their goals and requirements. In data-driven model building, there is a clear trade-off between accuracy and interpretability (Gleicher [2016]), even though visually debugging trained models can also help to identify problems and improve their performance (Liu et al. [2018]). The main objective is still to derive a model that predicts values accurately and generalizes well to new inputs, which means that accuracy is usually weighted more than interpretability.

In some use cases of predictive modeling, it is particularly important that automatic decisions can be trusted (Chatzimparmpas et al. [2020b]), which demands for interpretable solutions that enable analysts to validate the generalizability of models and take appropriate actions. In the medical domain, for instance, analysts need to understand why certain predictions were made “for determining targeted interventions” (Lundberg et al. [2018]). In recent years, there has been a stronger research focus on explainable artificial intelligence (XAI) (Barredo Arrieta et al. [2020]; Choo and Liu [2018]) that aims toward responsible AI, particularly in light of the increasing impact of automated decisions in life, from health to creditworthiness. Numerous approaches have been developed to visually analyze and debug big neural network-based models after training (Choo and Liu [2018]), for instance. Chatzimparmpas et al. [2020a] review several surveys about such visualization-based approaches for interpreting machine learning models.

However, the focus of AIX shifts toward understandable models, that is, the design of the model should already take interpretability into account. AIX aims to uncover interesting patterns and relationships in a specific data set and focuses less on encapsulating generalized concepts for predictions on unseen data. Developing powerful but also visually interpretable techniques benefits both fields, AIX and XAI. In the case of AIX, though, a slight loss in accuracy is acceptable since the model is merely a proxy for structuring and visually analyzing the data it was trained on or fitted to.

2.5.3 Applications of AIX

AIX describes a visual analytics process in which an interpretable model is trained on or fitted to a data set such that the visual analysis of said model may lead to insights about the data. The goal of this process is to incite and steer new AI-based methods for the large-scale analysis of textual and multivariate data, but it also helps to characterize existing workflows that deal with interpretable models for explaining data sets.

Multivariable linear regression is frequently used to infer which independent variables x_i exhibit the most (linear) influence on the dependent variable y by solving $y = \sum_i a_i x_i + b$ for the coefficients a_i (Schneider et al. [2010]). The resulting coefficients describe the linear relationships in the fitted data set and are easy to interpret. Hence, performing linear regression to extract and interpret the coefficients is a simple example of AIX. Similarly, Figure 2.12 shows that analysts can inspect decision trees not only to understand the decision-making process but also to learn more about combinations that correlate with a specific target outcome.

This thesis proposes several techniques to scale the visual analysis of multivariate and textual data sets using artificial intelligence and visualization. In Chapter 4, the AIX process is applied in the context of dimensionality reduction to generate context-aware, date-aware, and word order-aware tag maps. The idea is to first extract tags from a timestamped document collection (e.g., news reports), investigate their semantic and temporal relationships, and then optimize a loss function that evaluates how well the two-dimensional tag layout preserves these multivariate relationships. The model parameters then correspond to the tag locations on the map. Several penalties are introduced to regularize for visual interpretability. For instance, the tags should not overlap and they should convey their temporal evolution based on the triangular layout.

Chapter 5 proposes an efficient yet explainable dynamic clustering algorithm for documents that facilitates the online analysis of social media posts, which is presented in Chapter 6. The inner workings of the resulting clustering is continuously visualized so that analysts can interpret the main topic in each cluster and understand how the clustering evolves over time.

Chapter 7 introduces *Visual Neural Decomposition* to explain which combinations of variable ranges lead to high values of a specific target variable in the set. It is based on a neural network architecture that is trained to predict the target variable based on the remaining variables, but, instead of using these predictions, the model is then visually dissected such that analysts can infer relevant relationships. This chapter further introduces a novel regularization term for the neural network such that the resulting model is easier to interpret.

Chapter 3

Efficient Visual Document Collection Summarization

In visual document analysis it is often important for analysts to get an overview of large document collections. For example, one might want to know which topics are currently trending in news reports or on social media. One common collection summary strategy is to extract frequently appearing terms, which is fast and straightforward. The downside of this is that less informative terms often rank highly, and single words rarely provide sufficient context for analysts to understand what exactly is going on. More advanced approaches (e.g., based on recurrent neural networks) provide longer summaries, but either target individual texts or are not efficient enough for the visual analysis of large collections, especially in a streaming environment. Gambhir and Gupta [2017] survey various techniques for summarizing text and text collections.

The aim of this chapter is to propose novel summarization techniques that combine the efficiency and simplicity of single term ranking methods with the expressiveness of longer summaries. The first part presents ELSKE, an efficient algorithm to extract relevant keywords and longer keyphrases from both individual documents and document collections. The second part proposes a technique to extract blocks of phrases that often appear in a certain order, so-called *quotes*.

Whereas ELSKE belongs to the well-established category of keyword extraction algorithms (albeit with a focus on efficiency and collections), the quote extraction technique offers more detailed insights into large collections of short texts. Rather than extracting only contiguous phrases (like ELSKE), it tries to find frequent text patterns, possibly with omissions in-between. Based on this technique, a visual analytics system was developed for the hierarchical content analysis of social media posts.

3.1 Efficient Keyphrase Extraction (ELSKE)

Automatically extracting descriptive words (keywords) or phrases (keyphrases) from documents is important for a wide range of tasks, including document summarization and improved information retrieval in databases (Alami Merrouni et al. [2020]). Several graph-based (Mihalcea [2004]; Wan and Xiao [2008]; Bougouin and Boudin [2013]; Škrlj et al. [2019]), statistical-based (El-Beltagy and Rafea [2009]; Rose et al. [2010]; Campos et al. [2020]), and machine learning-based methods (Meng et al. [2017]; Xiong et al. [2020]; Wang et al. [2020]; Ye and Wang [2020]; Santosh et al. [2020]) have been developed to find a limited set of concise words or phrases that best describe a certain document.

According to Hasan and Ng [2010], a typical keyword extraction pipeline consists of two steps. First, the algorithm extracts a set of candidates. Then, a suitable ranking is applied to retrieve the best fits. Part-of-Speech (PoS) tagging is often used to retrieve candidates that are composed of nouns and adjectives (Hasan and Ng [2010]; Mihalcea [2004]; Wan and Xiao [2008]), but this excludes longer sequences. Furthermore, most PoS taggers need a considerable amount of processing time (Horsmann et al. [2015]). YAKE (Campos et al. [2020]) does not make use of PoS tagging, but focuses on extracting up to tri-grams per default. In recent years, machine learning techniques have been proposed that advanced the state of the art, but powerful models are computationally expensive, need extensive training data, and may generalize less to novel domains due to the supervised training.

This thesis focuses on large-scale visual analyses of collections, for instance, to gain an overview of recent news reports or trending developments on social media. However, existing methods largely focus on extracting keywords from individual documents. In addition, they often do not take into account the particular challenges of analyzing streaming data such as continuously incoming tweets. Dealing with large amounts of micro-documents such as tweets is challenging in several ways, particularly regarding information extraction tasks (Imran et al. [2015]).

Approaches to summarize documents often do not adapt well to very short texts. Short documents provide little context that can be harvested for mining descriptive keyphrases, and the sheer quantity of newly published items per second requires a great amount of computational resources or efficient methods. This is particularly important for applications that need to process incoming documents immediately, for instance, in scenarios that aim at providing situational awareness. The variety of spelling and lack of grammar further complicates the analysis. Dealing with large collections, we may also want to find frequent longer phrases to better understand the underlying data with additional context information.

Unfortunately, we can hardly rely on syntactical and structural assumptions for finding suitable phrase candidates if we choose to avoid Part-of-Speech tagging for efficiency. But without constraints, every subsequence with length $< m$ is a potential keyphrase candidate, and the set of distinct sequences grows prohibitively large for $m \gg 2$. Another issue is that an extended set of candidates will also lead to an increase of similarly worded keyphrases. Extracting context-rich keyphrases from large datasets in a timely manner is therefore particularly challenging.

Hence, this thesis proposes *ELSKE*, a new method to efficiently extract descriptive but potentially long phrases that appear unusually often, including complete sentences. The main idea is to first collect statistics about every uni- and bigram in the collection, which is then used to find a limited set of potential keyphrase candidates that may appear unusually often. For the ranking, this thesis extends the concept of TF-IDF to phrases and adapts it to the analysis of large document collections.

3.1.1 Method

In the following sections, the term *source* refers to the document or the collection of documents from which we want to extract keyphrases. For analyzing collections, the individual documents are concatenated into one big document. For instance, a system may continuously retrieve tweets, stores them in a sliding window, and then applies the algorithm at regular intervals to the source which is the concatenation of all tweets in the sliding window.

Let V be the vocabulary of terms in the source, then each keyphrase p^i is a sequence $(v^i_1, \ldots, v^i_m)$ with $v^i_j \in V$. Ranking candidate keyphrases with TF-IDF is often used as a baseline to evaluate more advanced ranking approaches, but it also performs surprisingly well in combination with Part-of-Speech tagging (Hasan and Ng [2010]; Meng et al. [2017]) and, importantly, has little requirements and external dependencies. Section 2.1.2 gives an overview of TF-IDF. The main idea of TF-IDF for ranking keywords is to weight terms v^i

with their frequency in the source $f_s(v^i)$ in relation to the document frequency of the term in a reference collection $f_d(v^i)$ comprising N documents:

$$\text{TF-IDF}(v^i) = f_s(v^i) \ln \frac{N}{f_d(v^i)} \tag{3.1}$$

One problem with this definition is that it is not immediately clear how this ranking could be extended to cover both single words and phrases. Hasan and Ng [2010] sum up the individual scores of each term to achieve this, but this favors long phrases. The proposed algorithm therefore sets the *phrase frequency* in the source in relation with the document frequency of the *phrase* in a reference collection. The document frequency is calculated on-the-fly using a document indexing structure of the reference collection, but it can also be approximated based on the document frequency of the rarest bigram in the phrase to save memory.

Unfortunately, with an increasing number of words in the source, the relation between phrase frequency and inverse document phrase frequency diverges and the influence of the latter one diminishes. For instance, in a diverse set of English news reports (e.g., from The Guardian), the maximum term frequency typically ranges around 500. If one analyzes the concatenation of thousands or even hundreds of thousands of documents, however, the most frequent term can easily appear more often than 10 000 times. Hence, this method introduces a sublinear scaling exponent $\frac{1}{\mu}$ to adapt the phrase frequency depending on the size of the source:

$$\text{PF-IDF}(p^i) = s(p^i) = f_s(p^i)^{\frac{1}{\mu}} \ln \frac{N}{f_d(p^i)} \tag{3.2}$$

If the maximum term frequency $f_s^{\max}$ exceeds 500, we set $\mu = \log_{500} f_s^{\max}$, otherwise $\mu = 1$. This means the algorithm scales the maximum term frequency in the source non-linearly down to the upper limit of 500, while keeping the scaled frequency of terms that only appear once at 1. In other words, we adjust the term or phrase frequency such that the typical relation between term frequency and inverse document frequency remains similar irrespectively of the size of the source collection.

Figure 3.1 gives an overview of the pipeline. The first four steps in the first block extract an initial set of candidate keyphrases. Then, three more steps in the second block filter these candidates to remove redundant and similarly phrased candidates.

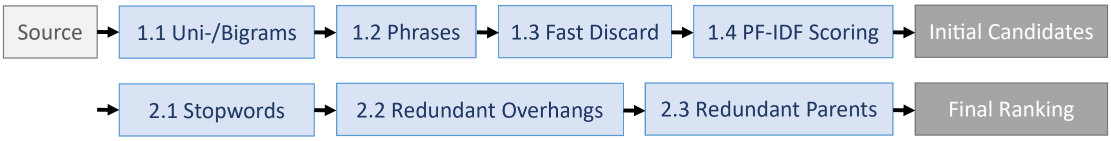


Figure 3.1 — The pipeline of ELSKE. An initial set of candidate keyphrases is extracted in the first block, which are then filtered in the second block to remove redundant and similarly phrased candidates.

Efficiently Extracting Candidate Phrases

With little assumptions about the resulting keyphrases, the number of theoretic candidates is huge. Extracting every $\{1, 2, \ldots, m\}$-gram in the source and computing the corresponding document frequency is not feasible, especially if the source comprises millions of sentences. However, we can exploit the fact that in most use cases the goal is to extract a limited number of k keyphrases, for instance, the top 1000. With this assumption we can speed up the process of extracting candidates as described in this section.

(1.1) Extracting Uni- And Bigrams

The algorithm first extracts uni- and bigrams, calculates their respective PF-IDF score $s(p^i)$, sorts the results in descending order, and stores the source and document frequencies in a map-like structure for fast retrieval. We exclude items that solely contain stop words.

The score at position k (s_k) is a lower limit (the list will only grow). Thus, one can divide s_k by the maximum possible inverse document frequency $\log N$ and raise it to the power of μ to retrieve the minimum frequency threshold f^{th}. It follows that every pattern in the final top k has to have a frequency of at least f^{th}. The algorithm only needs to extract phrases that appear at least as often as this computed threshold. The higher f^{th}, the higher the speedup.

(1.2) Extracting Longer Phrases

We now need to collect phrases that contain more than two words and calculate their frequencies. The algorithm ignores phrases that only contain stop words or appear only once (in case the minimum frequency is 1). A naive approach would need to look at and count every $\{1, 2, \ldots, m\}$-gram at every position in the source, leading to a squared time complexity of the two nested loops (the outer loop refers to the start position and the inner loop to the length m of the current m-gram). With the lower limit of f^{th}, though, we can stop the inner loop earlier if the frequency of the current (rightmost) bigram is below the threshold. The frequency of any bigram in a sequence is an upper limit of the frequency of that sequence and any longer sequence. The algorithm can retrieve the frequency of any bigram in $O(1)$ because we have already counted these in the first step. At the end of this step, we discard every phrase that does not meet the frequency threshold of f^{th}.

(1.3) Discarding Redundant Sub-Phrases

For each phrase $(v_1^i, \ldots, v_m^i)$, the algorithm will also return $m-1$ sub-phrases $(v_1^i), (v_1^i, v_2^i), \ldots$ as part of the candidate set from the previous steps. We discard such sub-phrases that have the same frequency in the source, because they only appear as part of the longer sequence.

(1.4) Calculating The Document Frequency

We need to retrieve the document frequency of each candidate phrase in our reference collection. To speed up the process, we can first build a bigram-based index of the collection. Another alternative is to estimate the document frequency with the document frequency of the rarest bigram in the phrase, which is an upper limit. Then, we can calculate the PF-IDF score of every phrase and discard patterns with a lower score than the threshold s_k.

Condensing The Set Of Candidates

After applying the first part of the pipeline, the algorithm could already extract the top k $\{1, 2, \ldots, m\}$-grams in the source according to the PF-IDF weighting scheme. However, among these candidates we often have several variations of similar phrases with slightly different frequencies. Hence, the field of candidates should be further condensed to retrieve the most salient and descriptive keyphrases.

(2.1) Stop Word-Heavy Candidates

We first remove *phrase* candidates if they only contain one term v_i that is not a stop word and $s(v_i) < s_k$. In these cases, only the additional stop word put the term above the threshold.

(2.2) Redundant Longer Candidates

Second, we remove longer phrases that provide little additional context. For instance, we discard *at a birthday party* if *birthday party* is one of the candidate phrases. We say a phrase p^j is a longer phrase of p^i if the sequence p^j contains the sequence p^i. For each phrase p^i, the algorithm determines whether there is a longer phrase p^j with at most two additional words in front of p^i and/or after p^i, the *overhang*. It only keeps the longer phrase if the individual PF-IDF score of any overhang is high enough (and does not contain only stop words), that is, $s(v_j) \geq \lambda s_k$ for an overhanging word v_j or $s((v_j, v_{j+1})) \geq \lambda s_k$ for an overhanging bigram. The default value is set to $\lambda = 0.1$. A lower lambda increases the

Ranking With Plain Phrase Frequency (TF-IDF):

trump (102388), **bernie** (44728), **hampshire** (22966), **bernie sanders** (20571), **roger stone** (14946), **primary** (17560), **doj** (17921), **people** (39007), **prosecutors** (14827)

Ranking With Adjusted Phrase Frequency (PF-IDF)

hampshire (22966), **nhprimary2020** (6365), **roger stone** (14946), **victory tonight is the beginning of the end for donald trump** (5065), **camden fairview high school** (4441), **hampshire primary** (4232), **stone case** (4152), **buttigieg** (8483), **bernie sanders** (20571)

Figure 3.2 — Top ten keyphrases from one million tweets published around Feb 12, 2020 (phrase frequency in brackets). The right depicts the final ranking using the adjusted phrase frequency (PF-IDF, $\mu \neq 1$), and the left using the plain phrase frequency (TF-IDF, $\mu = 1$)

number of additional phrase variations. If the overhang to the left or right is more than two words, the algorithm assumes that the longer phrase is unique enough compared to the shorter phrase.

(2.3) Redundant Shorter Candidates

Third, we discard shorter phrases that are already well represented by longer phrases. Given a candidate phrase p^i, we determine its set M^i of the shortest and distinct longer phrases among the candidates, that is, any phrase $p^j \in M$ must not be a longer phrase of any other $p^l \in M$ and must be *incompatible* with any other $p^l \in M$. A phrase p^j is *incompatible* with p^l if they share a common subsequence (p^i in this case), but continue differently in either direction. As an example, '*happy birthday*' is incompatible with '*great birthday*', but not with '*birthday party*'. We remove the phrase p_i if $s(p^i) - \sum_{j \in M^i} s(p^j) < s_k$. For instance, we would discard '*day*' if the candidates '*memorial day*' and '*st patricks day*' were already covering most occurrences of '*day*'.

Use Case

Most approaches try to find already good initial candidates so that they only need to rank these in the final step. In contrast to this, ELSKE first collects candidates in a broader way, performs an initial ranking, and then reduces the set of keyphrases for the final ranking.

The advantage of this strategy is that there are much less restrictions regarding potential keyphrases. The final list may contain phrases that start and/or end with stop words, as well as complete sentences. At the same time, the second part of the pipeline keeps the number of redundant phrases low.

Figure 3.2 depicts an example of the approach applied to one million tweets. Every tweet was converted to lowercase, emojis were removed, the # was stripped from hashtags, and the text was then tokenized into individual terms without punctuation characters. It shows that the top keyphrases contain both

Table 3.1 — Performance of the candidate selection process (time in seconds) compared to counting every {1,...,m}-gram (baseline).

	1k tweets		**100k tweets**		**1m tweets**	
	Time	Speed-Up	Time	Speed-Up	Time	Speed-Up
Baseline	0.896	1	49.16	1	642.28	1
Top k=100	0.009	100	0.59	83	9.76	66
Top k=1000	0.110	8	1.33	37	15.79	41

single terms and longer phrases, and that the sublinear scaling (PF-IDF) reduces the score of frequent terms that reveal little context.

3.1.2 Evaluation

Performance Comparison

The aim of this section is to quantify the speed-up of the phrase candidate extraction pipeline (step 1.1 to 1.4) compared to the baseline, that is, extracting and calculating the PF-IDF score of every {1,2,...,m}-gram in the source. To make the comparison fair, parallelization was disabled and the scripts used the same methods for both approaches to discard sub-phrases (step 1.3) and calculate the TF-IDF score (step 1.4), including the bigram-based index structure to quickly determine the document frequency in the reference collection.

The test script ran different configurations on a collection of lowercase tweets without punctuation. The results are based on the average duration of three runs for each configuration. The reference collection to determine the inverse document frequency comprises two million tweets. Each tweet is made up of 20 words on average.

Table 3.1 shows that the selection process is between one and two orders of magnitude faster than counting every m-gram. This does not include the run time of the second part of the pipeline, which approximately needs between 200 and 400ms for the third case of one million tweets and is thus negligible compared to the selection process. The script tokenizes the input and converts it to a vector-based representation for both approaches. This is comparable to the processing time of the $k = 100$ candidate selection process.

It should be noted that these results reflect the performance on a single core and the parallelized implementation scales well with additional cores. Furthermore, it typically takes even longer than this baseline to tag the same amount of data with a decent PoS tagger. For instance, the popular Stanford PoS tagger would need approximately between half an hour and five hours to process 20m tokens, depending on the model (Horsmann et al. [2015]).

Table 3.2 — Benchmarks on present keyphrase prediction with reported values from a) Meng et al. [2017] b) Chen et al. [2020] and c) Martinc et al. [2021]. The last two RNN-based methods are supervised. Among the unsupervised methods, ELSKE achieves better or similar results for most of the benchmark data sets (excluding Inspec).

	SemEval		**Krapivin**		**Inspec**		**NUS**	
	F_1@5	F_1@10	F_1@5	F_1@10	F_1@5	F_1@10	F_1@5	F_1@10
ELSKE	22.0	22.5	22.6	19.5	20.7	23.6	26.1	20.6
TF-IDF (PoS)[a)]	12.8	19.4	12.9	16.0	22.1	31.3	13.6	18.4
TextRank[a)]	17.6	18.7	18.9	16.2	22.3	28.1	19.5	19.6
SingleRank[a)]	13.5	17.6	18.9	16.2	21.4	30.6	14.0	17.3
ExpandRank[a)]	13.9	17.0	8.1	12.6	21.0	30.4	13.2	16.4
TopicRank[b)]	8.3	9.9	11.7	11.2	n/a	n/a	11.5	12.3
YAKE[c)]	15.1	21.2	21.5	19.6	20.4	22.3	15.9	19.6
CopyRNN[a)]	29.1	30.4	31.1	26.6	27.8	34.2	33.4	32.6
CorrRNN[b)]	32.0	32.0	31.8	27.8	n/a	n/a	35.8	33.0

Benchmarks

The main aim of ELSKE is to analyze collections of documents rather than single documents, but in order to compare it with previous work, its performance was evaluated on the SemEval (Kim et al. [2010]), Krapivin (Krapivin [2008]), Inspec (Hulth [2003]), and NUS (Nguyen and Kan [2007]) data sets. The procedure largely follows that of Meng et al. [2017] and Chen et al. [2020].

The test script analyzed titles and abstracts, and measured the $F_1@k$ scores of present keyphrases of the gold standard. The F_1 score combines precision and recall:

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{3.3}$$

The $F_1@k$ score refers to the F_1 score based on the first k results of the final ranking.

The test script applied stemming when comparing the extracted keyphrases with the gold standard and for determining which keyphrases in the gold standard are *present*, to make the results compatible with reported scores in related work. The script used the list of English stop words from the NLTK toolkit[1]. It should be noted that *TF-IDF (PoS)* describes the TF-IDF-based baseline method which uses (computationally expensive) PoS-based rules for retrieving candidates (Hasan and Ng [2010]).

[1] https://gist.github.com/sebleier/554280

The results in Table 3.2 show that while the *supervised* recurrent neural network-based techniques take the lead, ELSKE is competitive among the unsupervised methods. Inspec comprises very short abstracts, which might explain the lower-than-average performance on this data set.

In summary, ELSKE can efficiently analyze large collections and imposes little restrictions on the length and structure of keyphrases, but it also performs reasonably well if targeted at individual documents. The short run time even on larger collections is particularly relevant if continuously incoming data has to be analyzed in a timely manner, for instance, when extracting keyphrases from streamed documents in a sliding window at regular intervals.

3.2 Interactive Hierarchical Quote Extraction

Extracting popular opinions and statements from social media is of vital interest for many stakeholders, including journalists investigating developing stories, and agencies monitoring brand exposure. Social media can also be a useful source of information for emergency responders when eye-witnesses post their observations. While trending hashtags or keywords can give analysts an overview of popular topics, they do not convey more concrete concepts and statements. Conversely, displaying a selection of frequently shared and liked documents ignores the vast majority of content.

The previous section introduced ELSKE for summarizing large collections such as tweets using context-rich keyphrases. While the returned phrases are easier for analysts to interpret than individual words, it is still difficult to aggregate and extract frequently shared statements and opinions that are worded similarly but different enough to be recognized as a contiguous keyphrase.

This section introduces a new concept to visualize aggregated concepts that is inspired by the way how textual quotes are often shortened to convey the main idea. For instance, statements like '*I love the extremely talented Joe Doe*' and '*I have to tell you that I love the actor Joe Doe so much*' could be aggregated with the statement '*I love ... Joe Doe*'. Both partial phrases have little meaning in themselves, only the combination reveals a designated sentiment.

The aim of the proposed approach is to automatically extract such shortened quotes at varying levels of detail to aggregate a range of different micro-documents that talk about similar things, but are phrased slightly different. The concept assumes that similar posts share multiple chunks which are possibly scattered throughout the sentence. As a result, connecting said chunks should support analysts to better make sense of the content. The idea is that they can then iteratively dive into patterns to extract more detailed quotes regarding themes of interest, down to the level of individual posts.

3.2.1 Background

Several concepts have been proposed to visualize unstructured text content while preserving the linguistic structure in some way. Hu et al. [2017] visually aggregate social media posts with node-link diagrams that visualize which words often appear in a certain order. However, the level of aggregation leads to the effect that the visualization may imply a pattern that does not occur in the underlying data set, a caveat also shared by Van Ham et al. [2009]. Tree-based approaches visualizing how sentences continue (Wattenberg and Viégas [2008]; Culy and Lyding [2010]) are optimized for the analysis of certain seed queries and typically do not scale well for many items and variants, because every possible pathway is considered. Furthermore, these approaches make it difficult to quantify the popularity of patterns.

3.2.2 Quote Extraction

Each quote is a sequence of word subsequences. For instance, *'new york ... shooting'* represents all texts containing *'new york'* and *'shooting'* in that order, with arbitrary content before, after, and in-between. Extracting such patterns is not trivial due to the sheer number of theoretically possible variations.

The algorithm was designed to aggregate a collection of sentences or short texts such as tweets. It is threshold-based, that is, the algorithm will (initially) find patterns that occur at least as often as some adjustable threshold. If the analyst double-clicks on an item, more fine-grained quotes matching the clicked pattern are extracted using a lower threshold, all the way down to individual documents.

The algorithm first extracts all *single* subsequences that meet the adjustable occurrence threshold and do not consist entirely of stop words. The threshold allows us to implement this extraction efficiently using word and word pair occurrence statistics of the input data that the algorithm collects at the beginning. This is analogous to the pattern counting step of the ELSKE algorithm introduced in Section 3.1.1. We drop patterns that are contained in longer patterns but occur equally often. For instance, *'how old are'* is removed if it always occurs as part of the pattern *'how old are you'*.

For each resulting item, we then try to find patterns with an additional subsequence (chunk), for instance, *'new york ... shooting'* if *'shooting'* was the initial subsequence. To achieve this, the algorithm first determines for each subsequence the matching input items. It then collects the word and word pair occurrence statistics separately on each such group. The reason for this is that we can then again exploit the occurrence threshold for an efficient retrieval of frequent quotes (now with two subsequences). Redundant shorter quotes are again dropped.

We iteratively repeat this process with increasing numbers of subsequences, until no new pattern is found meeting the threshold. The algorithm then discards similar quotes by removing shorter quotes that occur similarly often as a matching longer quote.

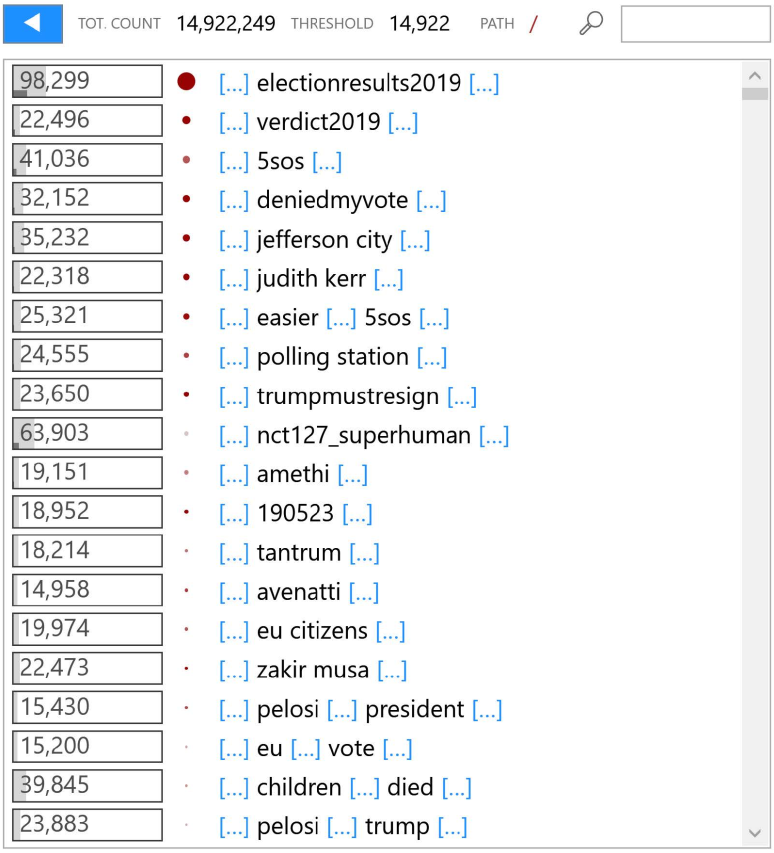
Analysts can select patterns to retrieve more fine-grained patterns. Then, the above process is started on the subset of the data that matches the pattern of interest, but with a lower threshold. The default occurrence threshold is one percent of the total number of documents for top level results, and five percent of the number of child items for lower-level results.

3.2.3 System Design

The system analyzes the loaded data set and visualizes the extracted patterns as depicted in Figure 3.3. Here, the top-level results of 15 million tweets published on May 23, 2019 are displayed. In this example, the results of a particular election, *'jefferson city'*, and *'judith kerr'* are among the most trending topics. Each row shows to the left the number of tweets matching the respective pattern. If several distinct posts back the same statement, we can assume that it is more likely to be relevant. The proportion of unique posts is therefore visualized with a thinner bar in darker gray to support analysts in determining credibility. A low proportion indicates that the respective quote is mainly sourced from many retweets of only one or a few original posts.

Quote Ranking

The algorithm may extract hundreds of patterns that appear more often than the specified occurrence threshold, but most of them are composed of rather generic words and short phrases that offer little insights. Hence, the quotes in the top-level view are ranked according to the number of associated unique posts and the likelihood of the word constellation. The idea is that unusually frequent appearances of text snippets with non-generic and less predictable content hint at an interesting and trending development. A simple language model estimates the likelihood that the respective word constellation would appear in a random collection of thousands of tweets. It is based on pairwise conditional word probabilities that have been derived from a reference corpus comprising millions of tweets.



TOT. COUNT 14,922,249 THRESHOLD 14,922 PATH /

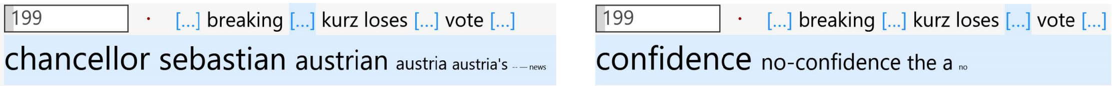
98,299	●	[...] electionresults2019 [...]
22,496	●	[...] verdict2019 [...]
41,036	●	[...] 5sos [...]
32,152	●	[...] deniedmyvote [...]
35,232	●	[...] jefferson city [...]
22,318	●	[...] judith kerr [...]
25,321	●	[...] easier [...] 5sos [...]
24,555	●	[...] polling station [...]
23,650	●	[...] trumpmustresign [...]
63,903	●	[...] nct127_superhuman [...]
19,151	●	[...] amethi [...]
18,952	●	[...] 190523 [...]
18,214	●	[...] tantrum [...]
14,958	●	[...] avenatti [...]
19,974	●	[...] eu citizens [...]
22,473	●	[...] zakir musa [...]
15,430	●	[...] pelosi [...] president [...]
15,200	●	[...] eu [...] vote [...]
39,845	●	[...] children [...] died [...]
23,883	●	[...] pelosi [...] trump [...]

Figure 3.3 — Main user interface with a ranking of the extracted top-level quotes from about 15 million tweets published on May 23, 2019. The number of matching posts is shown to the left of each quote and visualized with bars in light gray. The thin bar in dark gray indicates the proportion of unique posts.

The ranking value is mapped to the diameter of the circle in dark red right next to the quote on the left, and the estimated likelihood to the opacity of that circle. This enables analysts to better understand the reasoning behind the ranking. For instance, a big circle in dark red hints at a quote that is both unlikely and frequent, whereas a big circle with a lighter shade indicates that the respective quote was mainly ranked highly because of the sheer volume of the associated posts.

Sneak Peek

Each omission is represented with three blue dots in square brackets and users can click on them to sneak a peek what they are aggregating. A list with the most frequent terms between the respective blocks will pop up underneath the quote. The font size of each word corresponds to its relative frequency.



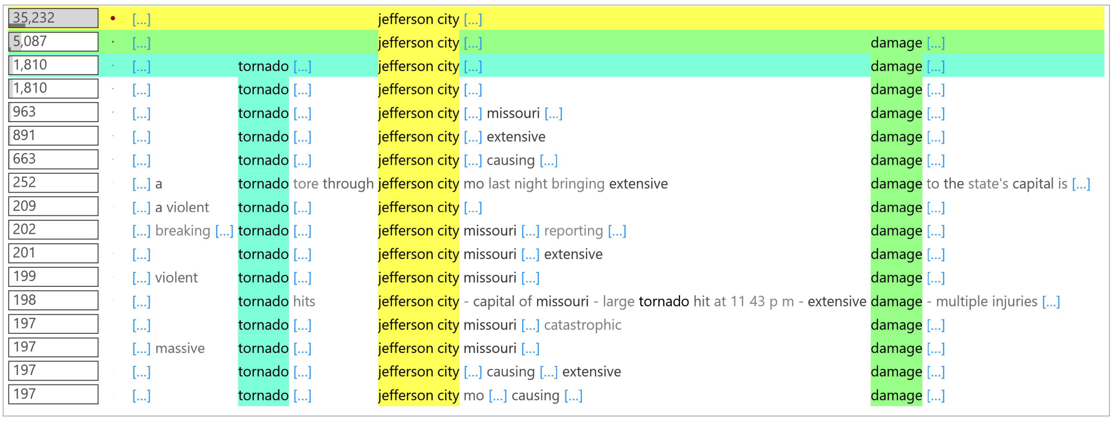
199 [...] breaking [...] kurz loses [...] vote [...]

chancellor sebastian austrian austria austria's -- — news

199 [...] breaking [...] kurz loses [...] vote [...]

confidence no-confidence the a no

Figure 3.4 — Sneak peek: users can click on an omission to reveal the most frequent terms between the blocks.



35,232	[...]		jefferson city [...]	
5,087	[...]		jefferson city [...]	damage [...]
1,810	[...]	tornado [...]	jefferson city [...]	damage [...]
1,810	[...]	tornado [...]	jefferson city [...]	damage [...]
963	[...]	tornado [...]	jefferson city [...] missouri [...]	damage [...]
891	[...]	tornado [...]	jefferson city [...] extensive	damage [...]
663	[...]	tornado [...]	jefferson city [...] causing [...]	damage [...]
252	[...] a	tornado tore through	jefferson city mo last night bringing extensive	damage to the state's capital is [...]
209	[...] a violent	tornado [...]	jefferson city [...]	damage [...]
202	[...] breaking [...]	tornado [...]	jefferson city missouri [...] reporting [...]	damage [...]
201	[...]	tornado [...]	jefferson city missouri [...] extensive	damage [...]
199	[...] violent	tornado [...]	jefferson city missouri [...]	damage [...]
198	[...]	tornado hits	jefferson city - capital of missouri - large tornado hit at 11 43 p m - extensive	damage - multiple injuries [...]
197	[...]	tornado [...]	jefferson city missouri [...] catastrophic	damage [...]
197	[...] massive	tornado [...]	jefferson city missouri [...]	damage [...]
197	[...]	tornado [...]	jefferson city [...] causing [...] extensive	damage [...]
197	[...]	tornado [...]	jefferson city mo [...] causing [...]	damage [...]

Figure 3.5 — The analyst has traveled down three layers to extract more fine-grained patterns matching *'tornado ... jefferson city ... damage'*.

Figure 3.4 depicts an example. Here, the sneak peek reveals that *'kurz'* refers to the then Austrian Chancellor Sebastian Kurz, and that the mentioned vote was a no-confidence vote.

Diving In

Analysts can double-click on any item to retrieve (longer) quotes concerning only the data that matches the clicked parent pattern. This parent pattern is then stuck to the top of the list with a distinct color. The related, more fine-grained quotes underneath are aligned with the parent, and the words matching the parent pattern are highlighted with the corresponding color. This way, the path of the analyst is visualized on their way down the tree. The highlighting conveys which parts of the quote contain new information, and the diversity of content between the text chunks becomes clear.

In contrast to the top-level result, the quotes in this view are ordered by their frequency. Analysts should be guided to the most common statements, and the likelihood that these statements are very generic is low since they are both longer and less frequent than the parent pattern.

In Figure 3.5, the analyst has first clicked on the pattern *'jefferson city'*, then on *'jefferson city ... damage'*, and finally on *'tornado ... jefferson city ... damage'*. The extracted quotes reveal that a violent tornado has apparently hit Jefferson City with extensive damage.

Analysts can also search for specific terms that are not part of the top-level list using the search box located at the top-right of the window. Then, popular quotes are extracted regarding all posts matching the query.

3.2.4 Use Case

In this use case, the approach was applied to the IEEE VAST 2019 Mini-Challenge 3 [2] to evaluate its utility. The goal of the challenge was to analyze social media messages from the fictitious city St. Himark using visual analytics to gain insights into issues related to a major earthquake that hit the city. For this challenge, the tool was slightly adapted to support the fictitious social media platform Y*INT of the challenge. Furthermore, additional views were implemented to visualize temporal and geographic data.

The provided data set contains about 42,000 posts, each with text, user name, and location. The system parses the location and converts it into the respective number of the city district. It further ignores 're:' prefixes indicating a 'retweet' to more accurately detect the number of unique posts.

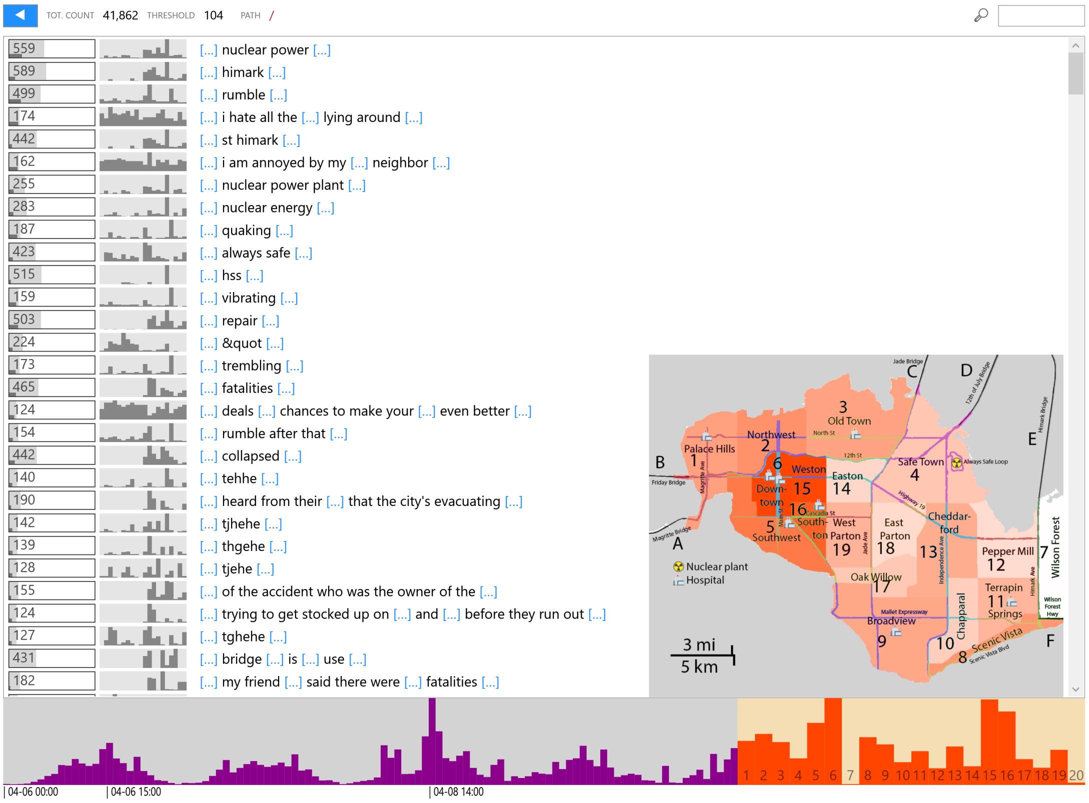
Overview

Figure 3.6 shows the initial view after loading the data set. On the Y*INT data set the ranking leads to some artifacts such as high-ranked misspellings of 'the' ('thgehe', 'tehhe', etc.). The reason for this is that the applied language model was built from Tweets and these variants rarely occur on real-life Twitter.

The slightly extended version of the system visualizes the number of matching posts over time with a small bar chart next to each quote. Additionally, there are two histograms on the bottom of the prototype. The left one (in magenta) displays the number of posts per hour regarding all items in the current hierarchy. The legend underneath reveals the date of important bars, namely the first one and peaks. To the right, the bar chart in dark-orange visualizes the number of posts per district. To make this more accessible, the user interface shades the corresponding districts on the city map from dark-orange (most posts per district) to white (no posts).

The views are linked, that is, if the analyst double-clicks on a pattern to retrieve more fine-grained quotes, the charts and map are updated accordingly to reflect the currently selected subset of the data. An example is given in Figure 3.7

[2] https://vast-challenge.github.io/2019/MC3.html



TOT. COUNT 41,862 THRESHOLD 104 PATH /

559	[...] nuclear power [...]
589	[...] himark [...]
499	[...] rumble [...]
174	[...] i hate all the [...] lying around [...]
442	[...] st himark [...]
162	[...] i am annoyed by my [...] neighbor [...]
255	[...] nuclear power plant [...]
283	[...] nuclear energy [...]
187	[...] quaking [...]
423	[...] always safe [...]
515	[...] hss [...]
159	[...] vibrating [...]
503	[...] repair [...]
224	[...] &quot [...]
173	[...] trembling [...]
465	[...] fatalities [...]
124	[...] deals [...] chances to make your [...] even better [...]
154	[...] rumble after that [...]
442	[...] collapsed [...]
140	[...] tehhe [...]
190	[...] heard from their [...] that the city's evacuating [...]
142	[...] tjhehe [...]
139	[...] thgehe [...]
128	[...] tjehe [...]
155	[...] of the accident who was the owner of the [...]
124	[...] trying to get stocked up on [...] and [...] before they run out [...]
127	[...] tghehe [...]
431	[...] bridge [...] is [...] use [...]
182	[...] my friend [...] said there were [...] fatalities [...]

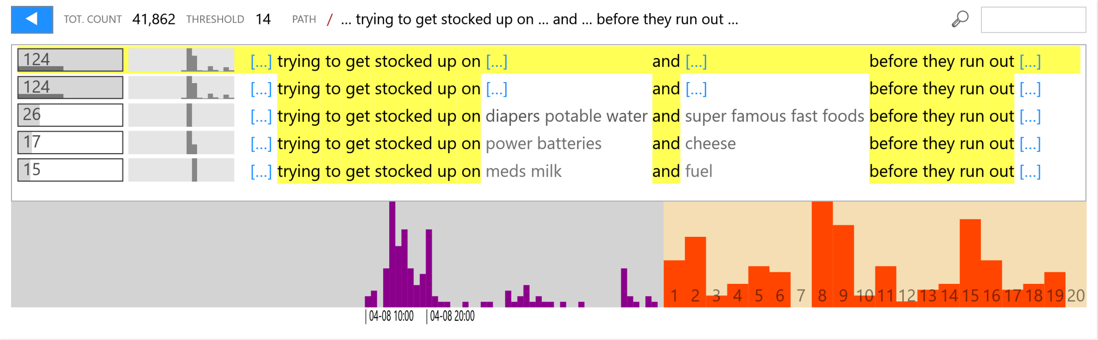
Figure 3.6 — Top-level quotes extracted from the Y*INT posts of the VAST Challenge 2019. Each row visualizes the temporal evolution and number of (unique) posts matching the respective pattern. The city map is shaded according to the location of the messages in the current view.

which shows the resulting view after the analyst double-clicked on the pattern mentioning food stocking.

Findings

The timeline and location bar chart in Figure 3.6 already reveal interesting aspects of the data set: the day-night patterns, some irregularities starting with the third day, that the most messages per hour were published on April 08 between 2 and 3pm, that hardly any message was sent from Wilson Forest, and that only very few messages are not associated with a location (the system associates them with the catch-all location *20*).

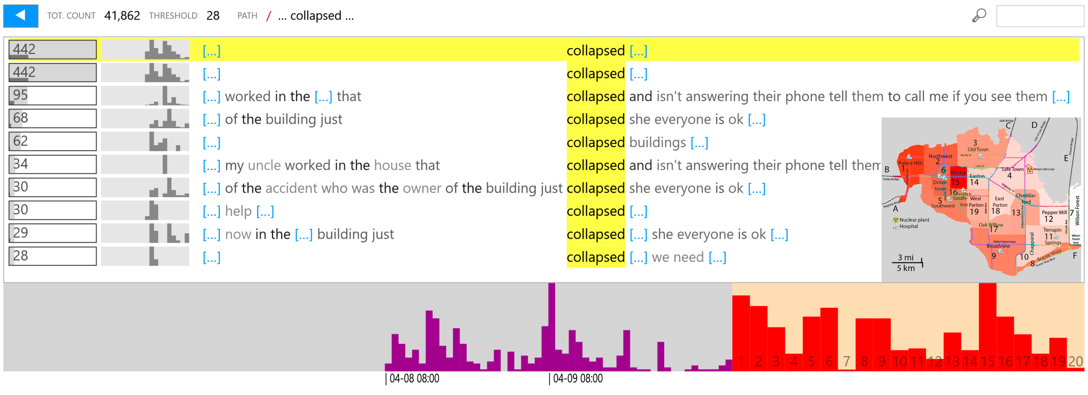
Looking at the top-level patterns, the analyst can deduce that there seems to be fatalities, people begin to panic about groceries, and the city is evacuating. Furthermore, many patterns suddenly appear during the second part of the



TOT. COUNT 41,862 THRESHOLD 14 PATH / ... trying to get stocked up on ... and ... before they run out ...

124	[...] trying to get stocked up on [...]	and [...]	before they run out [...]
124	[...] trying to get stocked up on [...]	and [...]	before they run out [...]
26	[...] trying to get stocked up on diapers potable water	and super famous fast foods	before they run out [...]
17	[...] trying to get stocked up on power batteries	and cheese	before they run out [...]
15	[...] trying to get stocked up on meds milk	and fuel	before they run out [...]

Figure 3.7 — The analyst has selected the pattern mentioning food stocking. The number of posts per hour at the bottom (in magenta) reveals that people begin to stock up groceries on April 8.



TOT. COUNT 41,862 THRESHOLD 28 PATH / ... collapsed ...

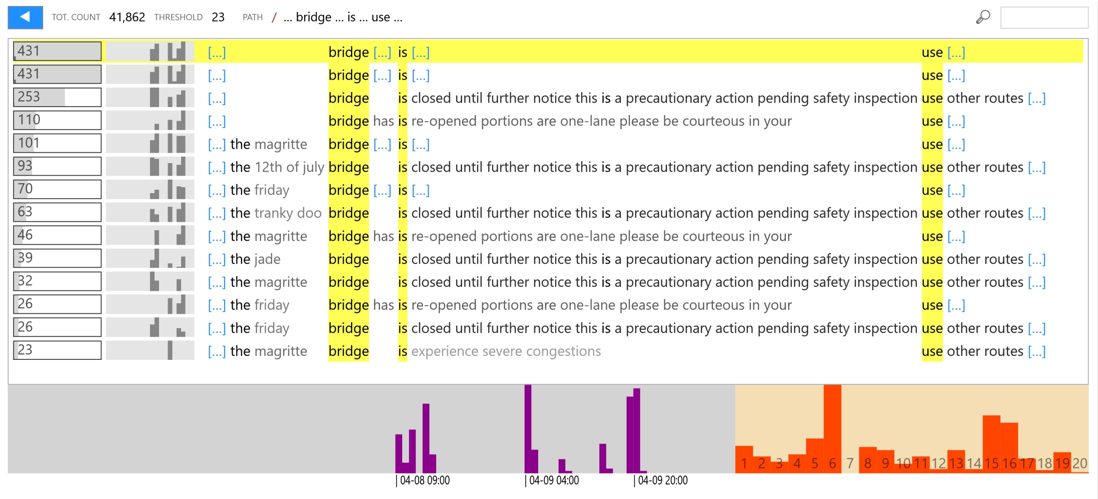
442	[...]	collapsed [...]
442	[...]	collapsed [...]
95	[...] worked in the [...] that	collapsed and isn't answering their phone tell them to call me if you see them [...]
68	[...] of the building just	collapsed she everyone is ok [...]
62	[...]	collapsed buildings [...]
34	[...] my uncle worked in the house that	collapsed and isn't answering their phone tell them
30	[...] of the accident who was the owner of the building just	collapsed she everyone is ok [...]
30	[...] help [...]	collapsed [...]
29	[...] now in the [...] building just	collapsed [...] she everyone is ok [...]
28	[...]	collapsed [...] we need [...]

Figure 3.8 — There are multiple reports regarding damaged buildings scattered throughout the city, and people worry about their relatives.

time range, for instance, *'collapsed'* or *'repair'*. Hence, something important seems to happen on the third day.

The analyst now wants to find out when the earthquake actually happened. Diving into the *'earthquake ... st himark'* pattern reveals warnings about a mild quake on April 6 that does not seem to be relevant here, and a major one with expected damage on April 8 at around 8am. Temporal patterns of posts containing *'collapsed'* (Figure 3.8) or mentioning food shortages (Figure 3.7) corroborate the hypothesis.

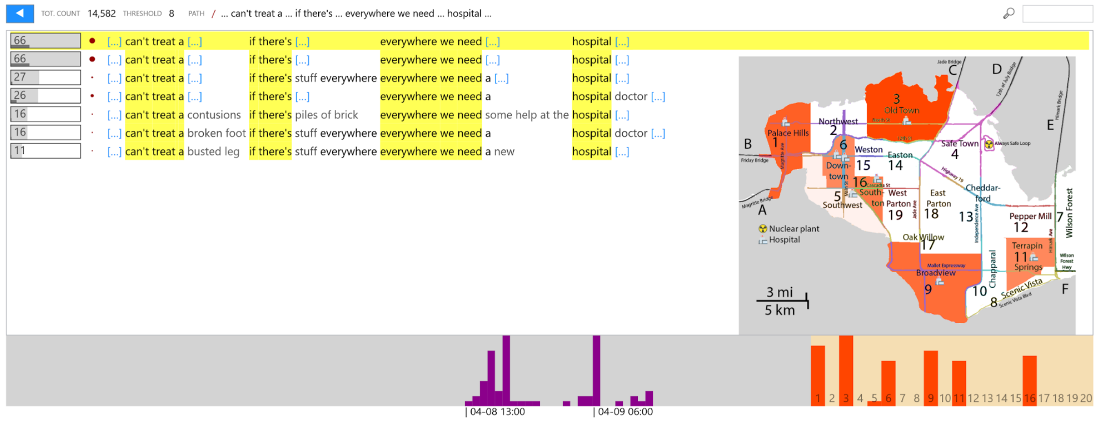
People complain about missing power and Figure 3.8 shows that there are many reports about collapsed buildings, especially north-west and around Downtown. Except for the Himark Bridge, every bridge is closed according to Figure 3.9, basically cutting off the island from the mainland.



TOT. COUNT 41,862 THRESHOLD 23 PATH / ... bridge ... is ... use ...

431	[...]	bridge [...]	is [...]		use [...]	
431	[...]	bridge [...]	is [...]		use [...]	
253	[...]	bridge	is	closed until further notice this is a precautionary action pending safety inspection	use	other routes [...]
110	[...]	bridge has	is	re-opened portions are one-lane please be courteous in your	use [...]	
101	[...] the magritte	bridge [...]	is [...]		use [...]	
93	[...] the 12th of july	bridge	is	closed until further notice this is a precautionary action pending safety inspection	use	other routes [...]
70	[...] the friday	bridge [...]	is [...]		use [...]	
63	[...] the tranky doo	bridge	is	closed until further notice this is a precautionary action pending safety inspection	use	other routes [...]
46	[...] the magritte	bridge has	is	re-opened portions are one-lane please be courteous in your	use [...]	
39	[...] the jade	bridge	is	closed until further notice this is a precautionary action pending safety inspection	use	other routes [...]
32	[...] the magritte	bridge	is	closed until further notice this is a precautionary action pending safety inspection	use	other routes [...]
26	[...] the friday	bridge has	is	re-opened portions are one-lane please be courteous in your	use [...]	
26	[...] the friday	bridge	is	closed until further notice this is a precautionary action pending safety inspection	use	other routes [...]
23	[...] the magritte	bridge	is	experience severe congestions	use	other routes [...]

Figure 3.9 — After the earthquake, most bridges seem to be closed, at least for several hours.

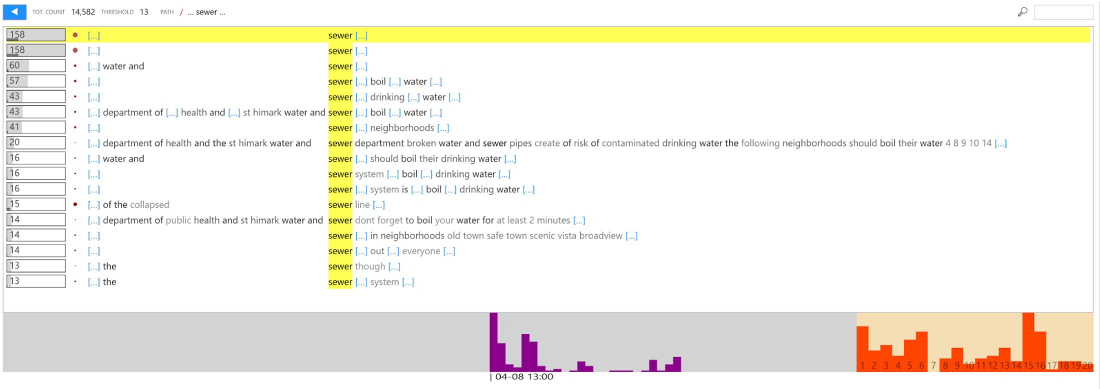


TOT. COUNT 14,582 THRESHOLD 8 PATH / ... can't treat a ... if there's ... everywhere we need ... hospital ...

66	●	[...]	can't treat a [...]	if there's [...]		everywhere we need [...]		hospital [...]	
66	●	[...]	can't treat a [...]	if there's [...]		everywhere we need [...]		hospital [...]	
27	·	[...]	can't treat a [...]	if there's	stuff everywhere	everywhere we need	a [...]	hospital [...]	
26	•	[...]	can't treat a [...]	if there's [...]		everywhere we need	a	hospital	doctor [...]
16	·	[...]	can't treat a	contusions	if there's	piles of brick	everywhere we need	some help at the	hospital [...]
16	·	[...]	can't treat a	broken foot	if there's	stuff everywhere	everywhere we need	a	hospital doctor [...]
11	·	[...]	can't treat a	busted leg	if there's	stuff everywhere	everywhere we need	a new	hospital [...]

Figure 3.10 — Hospitals have problems treating patients due to the damage that the earthquake has caused.

Browsing through the top ranked quotes, the analyst gains further insights into the situation after the earthquake. Figure 3.10 reveals that hospitals have troubles fulfilling their job due to piles of brick and *'running out of critical medical supplies'*. All hospitals seem to be affected according to the locations of the messages. Damage to the sewer system is reported around 5 hours after the earthquake (Figure 3.11) and several posts urge people to boil their drinking water.



TOT. COUNT 14,582 THRESHOLD 13 PATH / ... sewer ...

158	[...]	sewer [...]
158	[...]	sewer [...]
60	[...] water and	sewer [...]
57	[...]	sewer [...] boil [...] water [...]
43	[...]	sewer [...] drinking [...] water [...]
43	[...] department of [...] health and [...] st himark water and	sewer [...] boil [...] water [...]
41	[...]	sewer [...] neighborhoods [...]
20	[...] department of health and the st himark water and	sewer department broken water and sewer pipes create of risk of contaminated drinking water the following neighborhoods should boil their water 4 8 9 10 14 [...]
16	[...] water and	sewer [...] should boil their drinking water [...]
16	[...]	sewer system [...] boil [...] drinking water [...]
16	[...]	sewer [...] system is [...] boil [...] drinking water [...]
15	[...] of the collapsed	sewer line [...]
14	[...] department of public health and st himark water and	sewer dont forget to boil your water for at least 2 minutes [...]
14	[...]	sewer [...] in neighborhoods old town safe town scenic vista broadview [...]
14	[...]	sewer [...] out [...] everyone [...]
13	[...] the	sewer though [...]
13	[...] the	sewer [...] system [...]

Figure 3.11 — There are reports about broken sewer lines and that people should boil their water.

3.2.5 Discussion

Keyword-based approaches to summarize document collections without preserving the word order can densely compress big data sets, but at a cost of less interpretable results. The proposed interactive approach aims to visually summarize large collections of text snippets such as Tweets while finding a good compromise between interpretability, conciseness, and scalability. The idea is to extract unusually frequent sequences of text blocks, which better preserve the context compared to keyword- and keyphrase-based summaries.

The advantage of the proposed approach is that the possibly shortened quotes enable analysts to read the resulting summaries, supporting sense-making tasks. While it is slightly more verbose than showing just keywords, the system is still capable of aggregating large collections by omitting chunks of words in-between. Importantly, only patterns that actually occur in the data set are visualized, and the popularity of each pattern is exactly quantified. In addition, analysts do not have to provide a starting pattern, enabling the exploration of unknown collections.

The hierarchical approach not only reduces the processing time on interactions, it also helps to scale the interactive analysis to large data sets. Every selection step drastically reduces the number of relevant items, making the extraction of more fine-grained quotes with a lower occurrence threshold feasible. The prototypical implementation can handle millions of sentences. The initial processing of one million tweets takes about 20 seconds, and diving into one of the popular top-level results takes about five seconds. Users can also influence the depth of the hierarchy. Setting a lower threshold shows more detailed quotes already at the top level, which flattens the hierarchy.

One shortcoming of the approach is that lower-level results often introduce visual redundancies. In the case of quotes with many blocks, the alignment can also lead to very wide visualizations that are harder to grasp. On the one hand, preserving the word order helps to make sense of the content. On the other hand, it can also reduce the effectiveness of the aggregation when different patterns cover similar statements. Finally, analysts need to be aware that the extracted quotes have been generated automatically and thus are not necessarily as meaningful, complete, and accurate as human-generated summaries.

CHAPTER 4

Interactive Exploration of Large Document Collections

As discussed in Section 2.3, analysts from different fields have to deal with large document collections with the goal to get a general overview of the data, but also to find interesting aspects and stories, often with little a-priori knowledge about the content. Business analysts, for instance, have to constantly monitor news reports to react to specific developments and make informed decisions. Journalists investigating an unauthorized document leak usually need to process large amounts of textual data in a short amount of time, which is labour-intensive (Yimam et al. [2016]).

It is difficult to analyze large text collections automatically if no or little information is available on the contained documents. Interactive visualizations help to provide compact summaries of such data sets and can support analysts to study promising aspects in detail, that is, they enable analysts to *explore* the data set interactively. *Tag clouds* (or *word clouds*) are popular choices to visually summarize large collections, but a collection of the most important single terms can only provide a basic overview of the content at hand. The previous chapter introduced ELSKE to extract potentially longer keyphrases that provide more context-rich summaries of documents (Section 3.1). However, it remains challenging to visualize these phrases so that analysts can infer relationships between different phrases and examine the evolution of patterns over time.

Hence, this chapter introduces *PyramidTags*, a novel approach for summarizing large text collections visually. In contrast to previous work, PyramidTags in particular aims at creating an improved spatial representation that incorporates both temporal evolution and semantic relationship of visualized tags within the summarized document collection, without requiring animations. Single- and multi-word tags from a time-stamped document collection (e.g., news reports) are extracted and placed onto a 2D spatialization (a *tag map*). Related tags are placed nearby, the position on the map indicates corresponding date ranges, and the word order is preserved to stimulate longer phrases. Interaction possibilities enable analysts to further explore concepts, reveal relationships within the collection and retrieve relevant documents. As a result, it equips analysts with a visual starting point for interactive exploration to not only get an overview of the main terms and phrases of the corpus, but also to grasp important ideas and stories.

The main contribution of PyramidTags is three-fold: First, the initial visualization promotes the understanding of large collections through a context-aware, date-aware and word order-aware layout. Second, interaction mechanisms with visual cues and suggestions guide analysts to dive deeper into topics of interest and retrieve relevant documents. Third, the approach is highly scalable and applicable to several hundreds of thousands of news reports using different time spans.

4.1 Background

Tag clouds polarize to a cetain extent whether they are appropriate for analytical tasks (Halvey and Keane [2007]; Rivadeneira et al. [2007]; Viégas and Wattenberg [2008]). Hu et al. [2017] argue that traditional tag clouds often only convey basic concepts. They stress that analysts require longer, connected phrases to grasp more complete ideas. Sinclair and Cardew-Hall [2008] found out that tag clouds are considered useful for browsing and information discovery, but less for seeking specific information.

Several methods have been introduced in recent years to improve the analytical capabilities of tag-based approaches, such as adding interaction (Heimerl et al. [2014]), clustering tags semantically (Wu et al. [2011]), or animating the temporal evolution (Chi et al. [2015]). To quickly reveal clusters of terms in a matrix visualization, Chuang et al. [2012b] introduced a seriation technique that also preserves the natural reading order. It has further been shown that context-aware tag cloud layouts can improve the understanding of the underlying documents (Hearst et al. [2020]; Wang et al. [2014]). Section 2.3.1 provides a more thorough discussion of related tag cloud approaches.

In all these cases, though, the approaches mainly focus on bringing forward a specific advancement such as context-aware layouts *or* visualizing syntactical structures *or* conveying temporal information. Additionally, even though many approaches have dealt with context-awareness in the past, Hearst et al. [2020] claim that there is still a lack of automated tools that reliably produce semantically grouped word clouds. The placement strategy of PyramidTags is inspired by the Triangular Model to express time ranges without animations or interactive sliders. The Triangular Model was originally introduced by Van de Weghe et al. [2007] to visualize interval-based data, based on work from Kulpa [1997].

Other approaches for summarizing collections utilize topic modeling to visualize pre-computed clusters of contiguous themes, including an approach for visually analyzing social media posts in real-time that is presented in Chapter 6. It is more straightforward to visually express the temporal evolution and relationships of a limited number of topics, but it may limit the creative and unbiased exploration through the introduction of pre-defined boundaries.

4.2 PyramidTags

PyramidTags presents analysts a date-aware, context-aware and word order-aware overview of large time-stamped document collections to support interactive exploration, topic selection and document retrieval. This chapter mainly focuses on applying the approach to several hundreds of thousands of news articles, but the concept also generalizes to other text documents.

4.2.1 Objectives

PyramidTags applies ELSKE (Section 3.1) to extract important single- and multi-word tags from documents within a specified date range, for example a week or a month, and lays out these tags considering several objectives:

Context-Awareness (O1): Related tags as indicated by the underlying data should be placed nearby

Word Order-Awareness (O2): If the underlying data implies a certain word order then the placement of the affected tags should adhere to that order

Date-Awareness (O3): The placement should reveal at which date range the respective tags mainly appear in the underlying data

Each objective aims to improve the layout such that analysts gain a more thorough overview of the data set they want to explore. Placing related tags nearby (**O1**) helps analysts to better understand the content (Hearst et al. [2020]; Wang et al. [2014]). If tags regularly appear close to each other then they are considered as being related (Section 4.4.4).

Furthermore, the approach should preserve the word order of the most important pairs in the map (**O2**) to visualize more descriptive and complex concepts of the data with longer phrases (Chuang et al. [2012a,b]; Hu et al. [2017]), particularly if analysts hover over tags.

Finally, the triangular layout (**O3**) offers two major benefits for analysts. On the one hand, it instantly provides more details of the document collection by revealing in which date range tags are mainly mentioned. This typically corresponds to when events happened (in case of news reports, social media, diaries, or written protocols, for instance). On the other hand, one can argue that the triangular layout stimulates clusters of topics, because surrounding tags exhibit a similar date range in which they are mainly mentioned in the document collection. For instance, two tags which both only appear at one specific day are less likely to be related if the dates are far apart. The quantitative evaluation in Section 4.5 corroborates this: even if one completely ignores the context- (**O1**) and word order-awareness (**O2**) objectives, the triangular layout still results in a more context-aware placement of tags compared to a conventional, random layout.

Hence, PyramidTags enables not only an instant visual overview of prominent words and phrases in a big corpus similar to multi-word tag clouds, it also conveys semantic relationships between tags (**O1, O2**), linguistic structures (**O2**), and temporal patterns (**O3**), revealing internal structures of the data set in a novel, comprehensive way. It is a closely coupled process of data analytics, visualization and interaction, extracting statistical and temporal relationships in the data first, which are then visually presented in an interactive environment.

Optimizing for several objectives that often conflict with each other is a challenging task. The optimal position of tags according to their temporal pattern (**O3**) may lead to violations of **O1** and **O2**, for instance. Thus, one needs to establish a viable compromise between all three objectives, which is achieved here by optimizing an objective function.

The objective function incorporates the different criteria to evaluate the resulting layout. Optimizing this function can be viewed as a more constrained form of dimensionality reduction that does not just try to preserve the distance between data points. Rather, it specifies more explicitly how certain relationships (e.g., the temporal evolution) should be preserved. In context of the AIX process (Section 2.5), we fit a model (the objective function with the tag positions as parameters) to the data (tag relationships in the data set) and then visualize

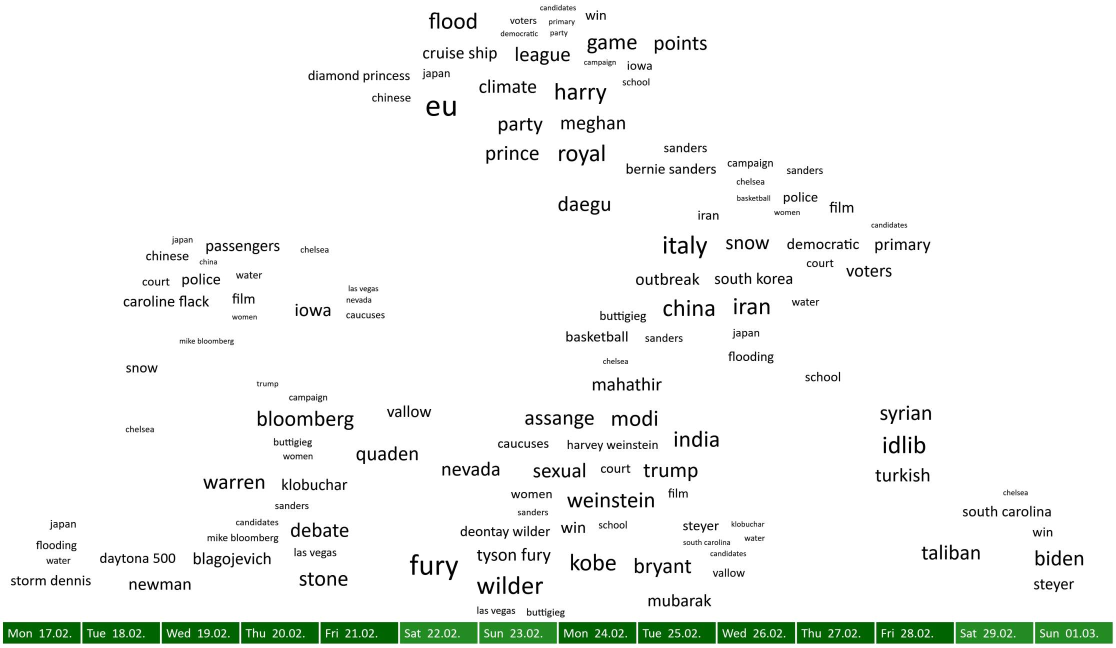


Figure 4.1 — PyramidTags visualization generated from about 70,000 news articles published in late January 2020.

the resulting model (tag positions). Additional regularization terms increase the visual interpretability of the resulting model (e.g., tags should not overlap). Section 4.4 describes these steps in more detail.

PyramidTags serves as a starting point to interactively explore large document collections. It provides analysts with several interaction mechanisms to reveal relationships in detail, select topics of interest, and retrieve relevant documents.

4.2.2 Overview

Figure 4.1 shows an example with 80 distinct tags that is based on about 70,000 news articles published in late January 2020. There is a timeline at the bottom of the visualization. Analysts can toggle whether spaces are replaced with underscores to better discern multi-word tags from accidental alignments

The vertical position of a tag indicates its temporal extent (duration), and the horizontal position the mid-point of the time range in which this tag mainly appears in the data. Tags placed at the bottom, right above the timeline, mainly occur on the specific day that is shown underneath. The higher a tag is placed, the longer its corresponding time range, that is, tags at the top are consistently

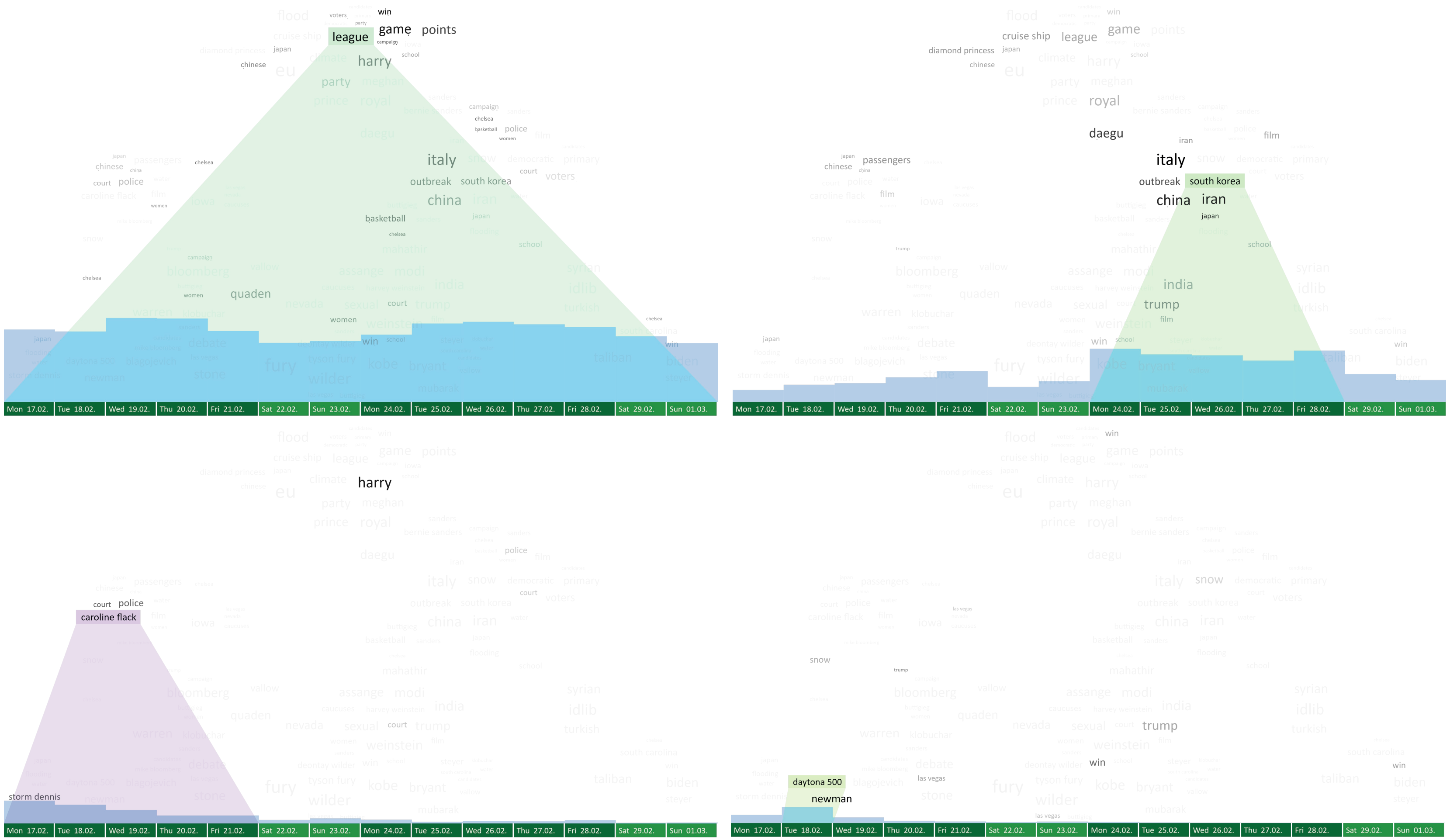


Figure 4.2 — PyramidTags applies the triangular layout to convey the temporal evolution of tags. In each of the four figures (based on Figure 4.1), the analyst hovers over a different tag of interest. The trapezoid beneath each tag indicates the prevalent date range of this tag in the data set, while the remaining tags are shaded according to their semantic connection with the tag selection. The bar chart at the bottom above the timeline visualizes the number of documents containing the selected tags per day.

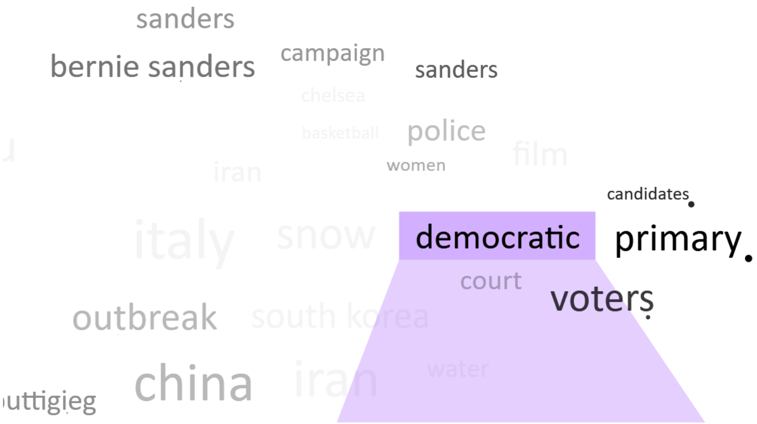


Figure 4.3 — User is hovering over the tag *'democratic'*. The other tags are shaded according to their relatedness. Little dots appearing under some tags reveal the word order that was determined in regard to the hovered tag.

mentioned throughout the entire processed data set. In contrast to simple time-to-space mappings, this placement strategy also visualizes data associated with intervals of time (time spans). It should be noted that the layout does not necessarily imply a topic hierarchy.

Figure 4.2 illustrates the placement strategy to convey the temporal evolution. In this example, the analyst hovers over different tags of interest. The trapezoid beneath each tag indicates the prevalent date range of this tag in the data set, while the remaining tags are shaded according to their semantic connection with the tag selection. For instance, it becomes clear that reports about the football league appear just as often throughout the two weeks (*'league'* is placed at the very top), whereas the news about Caroline Flack's death peak in the first few days (*'caroline flack'* is placed in the center and to the left).

4.2.3 Hovering Tags

When analysts hover over a tag, the remaining tags are shaded depending on how related they are to that tag, from black (very related) to nearly-white (data does not suggest relation). Figure 4.3 shows an example of this behavior. While related tags should be placed nearby, not every tag i that is placed next to tag j is actually related to it, and in some cases there could even be a strong connection of j to a different tag h that is placed much further away. Hovering over terms enables analysts to debunk false friends and sense how strong the connection really is. For instance, *'snow'* appears right next to *'democratic'* in Figure 4.1, but the nearly invisible shading in Figure 4.3 reveals that these terms do not frequently occur close to each other in the data set.

In addition, a little dot under each term appears if the data implies an ordering regarding the currently hovered tag. The interface uses circles to encode both the direction and the manifestation of the word order in the data set. The horizontal position of the dot depends on the percentage of occurrences in the collection with the indicated order. If the dot is placed to the left, that tag mainly appeared before the tag which is currently hovered. Analogously, if it is on the right, then the tag mainly appeared after the tag of interest in the document collection. The size of the dot indicates how sure the system is that there is a suggested word order, that is, how often the respective pair occurs in that order. These hints tell users whether the order of the tags as it is shown on screen is relevant and consistent with the statistics from the underlying data. For instance, the term *'candidates'* probably appears often after *'democratic'* and not the other way round if both are closely mentioned (Figure 4.3).

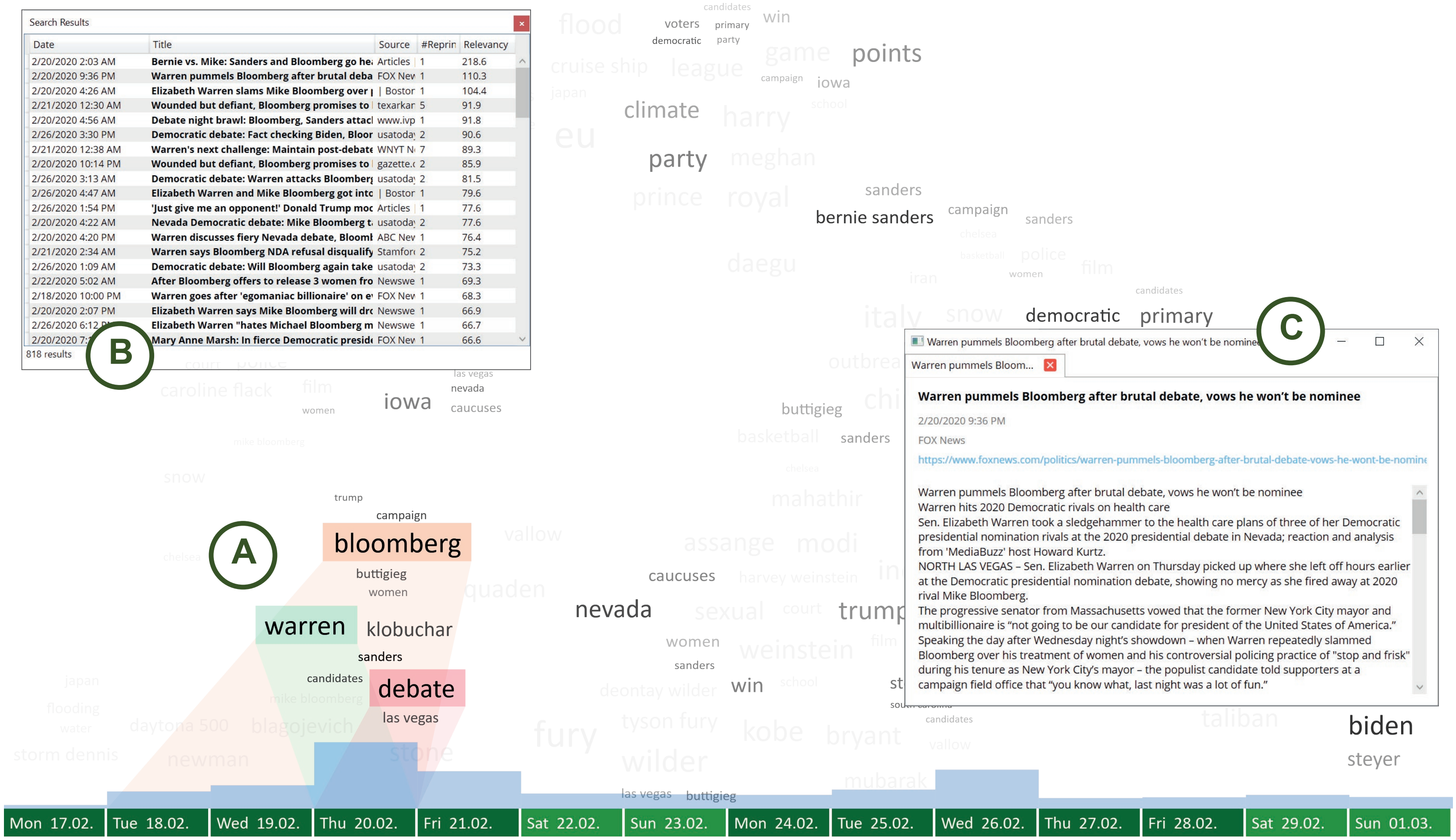
Each tag is associated to a specific time range during which it was strongly mentioned in the corpus. When hovered, this range is illustrated with a trapezoid that spans from the tag to the start and end date on the bottom at the timeline. Furthermore, a more detailed view regarding the temporal evolution of the hovered tag is presented with a bar chart popping up right above the timeline. Each bar sitting on its date is mapped to the corresponding number of documents the tag of interest appeared in on that date.

Figure 4.3 shows the updated visualization when hovering over the term *'democratic'*. The shading of the surrounding terms indicate several strongly related tags such as *'primary'*, *'voters'*, and *'candidates'*. The associated time range reveals that *'democratic'* was particularly often in the news in the second part of the time window. The dots under some tags reveal the assumed word order. For instance, one could read *'democratic ... primary'*, or *'democratic ... candidates'*. This suggests that there were many reports about the democratic primary election and debates in late January 2020.

Not every related tag i is necessarily placed close to the tag of interest j, especially if there is another tag k that has a strong connection to i. For instance, hovering over *'democratic'* (Figure 4.3) also highlights the terms *'biden'* and *'south carolina'* (Figure 4.1 bottom-right), but these terms are placed around March 1st when it was announced that Joe Biden had won the democratic primary election in South Carolina.

4.2.4 Multiple Tag Selection and Document Retrieval

Analysts can select and deselect several tags by clicking on them. Then, the selected tags with their corresponding triangles remain highlighted. The opacity



Search Results

Date	Title	Source	#Reprin	Relevancy
2/20/2020 2:03 AM	Bernie vs. Mike: Sanders and Bloomberg go he	Articles |	1	218.6
2/20/2020 9:36 PM	Warren pummels Bloomberg after brutal deba	FOX Nev	1	110.3
2/20/2020 4:26 AM	Elizabeth Warren slams Mike Bloomberg over	| Bostor	1	104.4
2/21/2020 12:30 AM	Wounded but defiant, Bloomberg promises to	texarkar	5	91.9
2/20/2020 4:56 AM	Debate night brawl: Bloomberg, Sanders attacl	www.ivp	1	91.8
2/26/2020 3:30 PM	Democratic debate: Fact checking Biden, Bloor	usatoday	2	90.6
2/21/2020 12:38 AM	Warren's next challenge: Maintain post-debate	WNYT N	7	89.3
2/20/2020 10:14 PM	Wounded but defiant, Bloomberg promises to	gazette.c	2	85.9
2/26/2020 3:13 AM	Democratic debate: Warren attacks Bloomberg	usatoday	2	81.5
2/26/2020 4:47 AM	Elizabeth Warren and Mike Bloomberg got into	| Bostor	1	79.6
2/26/2020 1:54 PM	'Just give me an opponent!' Donald Trump moc	Articles |	1	77.6
2/20/2020 4:22 AM	Nevada Democratic debate: Mike Bloomberg t	usatoday	2	77.6
2/20/2020 4:20 PM	Warren discusses fiery Nevada debate, Blooml	ABC Nev	1	76.4
2/21/2020 2:34 AM	Warren says Bloomberg NDA refusal disqualify	Stamforc	2	75.2
2/26/2020 1:09 AM	Democratic debate: Will Bloomberg again take	usatoday	2	73.3
2/22/2020 5:02 AM	After Bloomberg offers to release 3 women fro	Newswe	1	69.3
2/18/2020 10:00 PM	Warren goes after 'egomaniac billionaire' on e	FOX Nev	1	68.3
2/20/2020 2:07 PM	Elizabeth Warren says Mike Bloomberg will drc	Newswe	1	66.9
2/26/2020 6:12 P	Elizabeth Warren "hates Michael Bloomberg m	Newswe	1	66.7
2/20/2020 7:	Mary Anne Marsh: In fierce Democratic preside	FOX Nev	1	66.6

818 results

Warren pummels Bloomberg after brutal debate, vows he won't be nominee

Warren pummels Bloom...

Warren pummels Bloomberg after brutal debate, vows he won't be nominee

2/20/2020 9:36 PM

FOX News

https://www.foxnews.com/politics/warren-pummels-bloomberg-after-brutal-debate-vows-he-wont-be-nomine

Warren pummels Bloomberg after brutal debate, vows he won't be nominee
Warren hits 2020 Democratic rivals on health care
Sen. Elizabeth Warren took a sledgehammer to the health care plans of three of her Democratic presidential nomination rivals at the 2020 presidential debate in Nevada; reaction and analysis from 'MediaBuzz' host Howard Kurtz.
NORTH LAS VEGAS – Sen. Elizabeth Warren on Thursday picked up where she left off hours earlier at the Democratic presidential nomination debate, showing no mercy as she fired away at 2020 rival Mike Bloomberg.
The progressive senator from Massachusetts vowed that the former New York City mayor and multibillionaire is "not going to be our candidate for president of the United States of America."
Speaking the day after Wednesday night's showdown – when Warren repeatedly slammed Bloomberg over his treatment of women and his controversial policing practice of "stop and frisk" during his tenure as New York City's mayor – the populist candidate told supporters at a campaign field office that "you know what, last night was a lot of fun."

Figure 4.4 — The analyst has selected three tags that are highlighted (A). The blue bar chart above the timeline depicts the number of documents containing all selected terms. The system ranks relevant articles (B), which can also be retrieved in full (C).

of the other tags is updated to reflect the lowest relatedness to any of the selected tags, which is an upper bound of the actual relatedness to the selection (computing the relatedness of all possible combinations in advance is not feasible). Analogously, the bar chart on the bottom is updated with the number of documents containing all selected terms for each day. Hence, analysts are supported in picking a topic they would like to explore more deeply. Suitable selections are suggested by highlighting relevant tags according to the document collection. In many cases, analysts have to select only few tags to greatly narrow down the search results even in big data sets comprised of millions of documents.

If analysts select one or several tags, a separate window opens presenting a ranked list of the most relevant news articles as specified by the selection. The list contains the date, title and source of the documents as well as the number of reprints (i.e., how many different news outlets published the same story) and the relevancy as determined by the system. Only documents that contain all the selected tags are shown in the list. The number of occurrences but also the distances between the tags in the document influence the relevance score. Documents are considered to be more relevant if the selected tags occur closer to each other rather than being scattered throughout the document, because this indicates that they are semantically connected in the document. The total number of documents matching the selection is shown at the bottom of the window.

Figure 4.4 shows an example. In the depicted case, the analyst has decided to learn more about the democratic primary debates in the data collection by clicking on *'warren'*, *'bloomberg'*, and *'debate'* (A). As expected, reports and opinion articles appear as to which of the candidates performed best (B), and several related tags (e.g., *'nevada'*) are shaded in dark gray. Full articles can be retrieved by double clicking on the respective list item. The document explorer shows the requested article in a new tab, revealing the full text and the link to the source page among other meta data (C).

4.3 Preprocessing and Data Analysis

PyramidTags is built on a pipeline with three main steps. First, the data at hand is algorithmically analyzed and relationships are extracted. Then, based on the collected statistics the actual visualization is created as explained in Section 4.4. Finally, the proposed visual analytics approach displays the generated visualization and provides several interaction possibilities as described in the previous section.

Technically, the method works with any collection of timestamped documents. However, it aims to visualize content relating to events that span different time ranges, which is typical for social media posts or news articles. Here, PyramidTags was applied to hundreds of thousands of online news articles collected from news websites. This data is particularly challenging due to the size but also due to the fact that it contains noisy, real-world articles crawled from the web.

4.3.1 Cleaning and Reprint Detection

Web-crawled articles often contain additional content that is not part of the actual article, for instance, links to related news articles or advertisements. The crawler strips paragraphs from the document if several other articles from this news outlet also contain the same paragraph, assuming that only text that is specific to this document is considered as useful content. If a cleaned article largely contains the same content as a previously processed article, but it is from a different source, then it is considered to be a reprint. If it is from the same source we discard it as duplicate. Reprints are still considered as being part of the corpus for subsequent data analytics and the visualization, because the decision of news sites which agency reports they distribute is an indication for the importance of the content. The reprint detection is mainly used to save computing resources and to improve the usability of the news retrieval scenario.

4.3.2 Tag Relationship Analysis

The goal of the system is to visualize large text corpora while preserving important relationships and structures within documents. To achieve this, the system extracts relevant tags with ELSKE (Section 3.1) and analyzes their relationship to each other. Tags that often appear together in documents should also be closer to each other in the visualization, tags that often appear in a certain order should also be ordered that way in the resulting visualization. Furthermore, the temporal evolution should be visible, for example, if several tags mainly occur in a specific time range.

After extracting the tags, the system needs to process the input data set again to analyze the relationship between tags. For each document, we first search for the tags that were extracted in the previous step and note their position in the text. For each search result, we increment the count for the specific tag and date which we need later on. Then, we look at each pair (t_i^p, t_j^q) in that document where p and q denote the respective positions in the text of tag t_i

and t_j, and t_i has a lower lexicographic rank than t_j to avoid counting the same pair twice.

We disregard this match if there is another match (t_i^p, t_j^s) with $|s - p| < |q - p|$ or (t_i^s, t_j^q) with $|q - s| < |q - p|$. This means, if we replace one tag of this pair with the same tag but at a different location, this must not lead to a closer pairing. For example, the pair *'[John] Doe was seen outside. [Doe] wore a black jacket'* is ignored, since there is a closer pairing with one of the tags involved: *'[John] [Doe] was seen outside...'*.

If all these conditions are fulfilled, we calculate the distance weight d^w. We define the distance weight as the inverse distance between the two matches plus one:

$$d^w(t_i^p, t_j^q) = \begin{cases} \frac{1}{1+q-p-\text{numWords}(t_i)} & \text{if } q > p \\ \frac{1}{1+p-q-\text{numWords}(t_j)} & \text{otherwise} \end{cases}$$

The distance weight d^w is added to the order-aware tag distance weight w_{ij} if $q > p$ and to w_{ji} if $p > q$. We also add the distance weight to the distance weight count for the specific pairing and date. The order-aware tag distance weight w_{xy} is then the sum of the inverse distances of valid tag pair matches (t_x, t_y) where t_x appeared before t_y.

After processing all documents, $w_{ij} + w_{ji}$ indicates how often the tags t_i and t_j appeared nearby in the data. That is, the higher the sum, the more related the terms are to one another. If one summand is high compared to the other one, we can assume that the underlying documents suggest a specific word order, for instance, *'john doe'* is more likely than *'doe john'*.

4.4 Visualization Generation

A good PyramidTags visualization should fulfill several training objectives. The location on the map should correspond with the associated time range (*location*), tags should not overlap (*collision*, *repelling*), related tags should be placed close to each other (*proximity*), and the natural reading order should be respected (*wordOrder*). Unfortunately, these objectives often contradict each other. For instance, if all tags are placed at the same position, we would achieve the best result concerning the context-awareness, that is, related tags are indeed close together. However, this drastically violates the collision objective that tags should not overlap each other. Hence, we need a viable trade-off between these objectives.

The following objective function f_θ represents all training objectives with parameters θ (the two-dimensional location of each tag on the map) as input:

$$\begin{aligned} f_\theta = &\lambda_1 \text{location} + \lambda_2 \text{collision} + \lambda_3 \text{proximity} \\ &+ \lambda_4 \text{repelling} + \lambda_5 \text{wordOrder} \end{aligned} \tag{4.1}$$

The goal is to minimize this function to retrieve a good set of parameters so that the resulting visualization adheres to all three main objectives. The individual objectives are weighted with the meta-parameters λ_x, allowing to balance the importance between them. For instance, if we set $\lambda_3 = \lambda_4 = \lambda_5 = 0$ and only have one location box spanning the entire image, then the optimization routine would lead to traditional dense (multi-word) tag clouds. The following sections describe in detail how the individual components of the objective function are composed of.

4.4.1 Layout and Map Locations

We set the font size of each tag so that its area is approximately proportional to the respective importance weight assigned by the ELSKE algorithm, which is denoted with weightedCount(t_x) in the following sections.

The aim of the approach is to visually encode the evolution of certain tags and their relationship over time. Based on the triangular layout, each tag is mapped to a specific *location box* that indicates when and for how long this tag mainly appeared in the underlying data set. On the bottom of the visualization, a timeline shows the dates within the processed time range of n_d days. There are n_d location boxes on the first row right above the timeline, representing each day. If a tag is mapped to one of these boxes on the first row, it mainly occurs on that day. The second row (counted from the bottom) is comprised of $n_d - 1$ location boxes that represent durations of two days. This can be continued until the top row is reached with one location box in which the tags associated with it span the whole n_d days. The resulting structure resembles a pyramid-like shape in 2D, hence the name PyramidTags. An example of this structure is shown in Figure 4.5, in which each non-empty location box (i.e., at least one tag is mapped to the box) is drawn as a rectangle.

The rows do not necessarily have equal heights, this is determined based on the space requirements of the most occupied box of that row. The width of the boxes increases with each row from bottom to top, because there are fewer boxes the tags are distributed to. However, the width does not grow linearly with the number of days the boxes represent. As a result, the box in the top row does not occupy the entire horizontal space that is available to preserve the pyramid metaphor. The analyst should still be able to guess the probable time range of each tag, even with the static visualization, by spanning a right-angled triangle

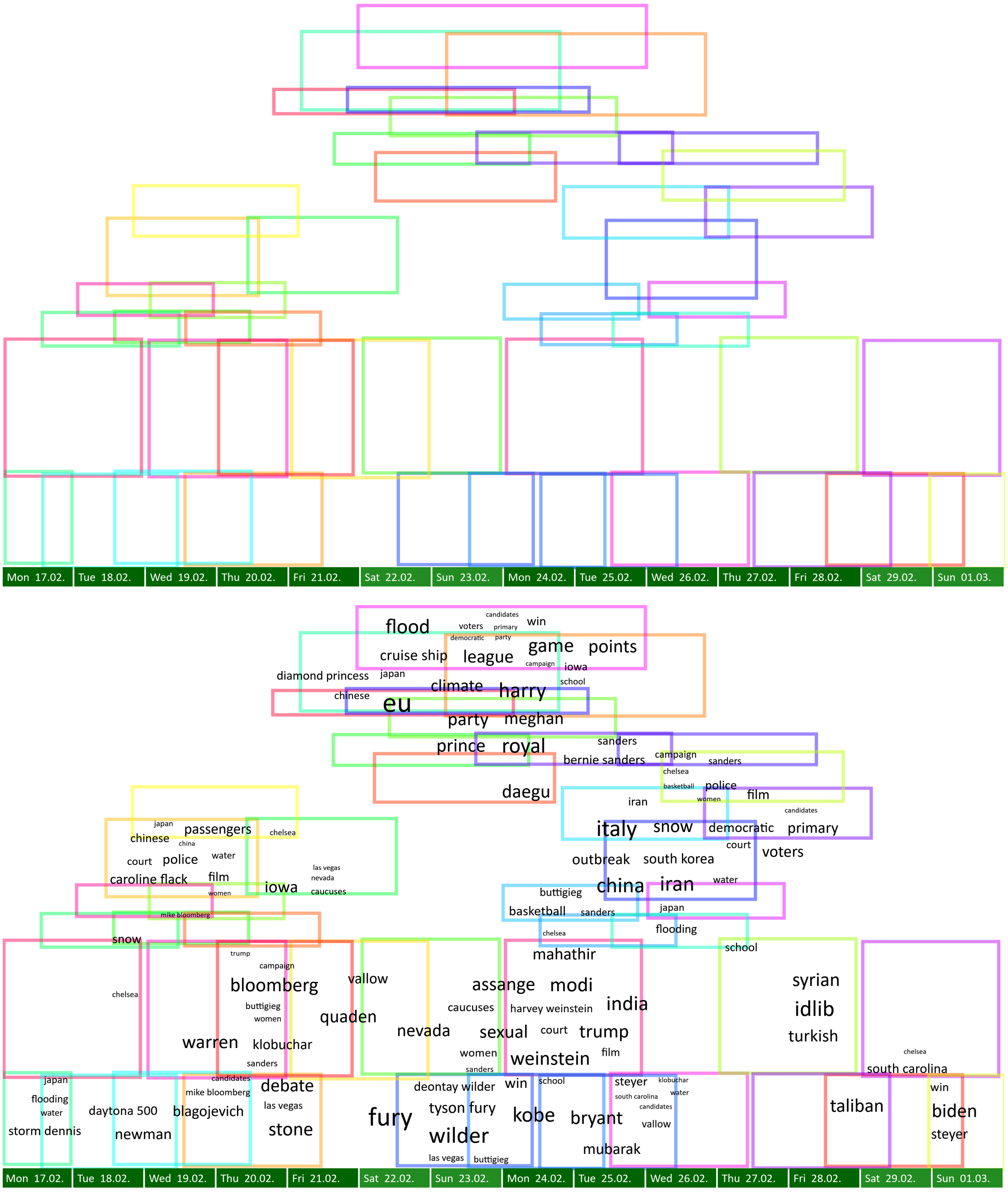


Figure 4.5 — Non-empty location boxes of the PyramidTags visualization from Figure 4.1. **Top:** location boxes with the tags hidden. **Bottom:** location boxes blended in with the resulting map. The objective function pushes each tag to its assigned location box indicating the prevalent date range. These boxes have dynamic widths and heights to optimize space usage while trying to preserve the triangle shape. Here, colors have been assigned randomly to location boxes to make them discernible.

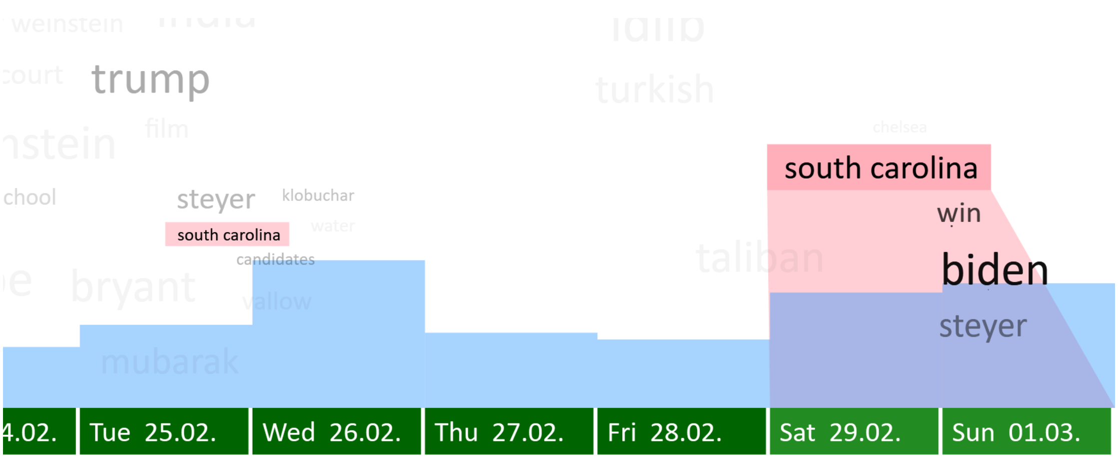


Figure 4.6 — Analysts hovers over the tag *'south carolina'* to the right. Another *'south carolina'* tag is highlighted on the left. The bar chart above the timeline reveals that there are two distinct peaks in the data set, therefore, the tag was split such that each peak can be assigned to a tag.

from the mid-point of the tag to the timeline at the bottom. The dynamic sizing is clearly visible in Figure 4.5. Each rectangle in the same row also has the same height, however, the height differs among rows.

Figure 4.5 shows that rows and columns can overlap each other, particularly with increasing number of total days. This is necessary to fit regular tags within the bottom row for large n_d. The flexible, dynamic approach for determining the height and width of location boxes optimizes the use of white space and preserves the pyramid-like structure at the same time.

4.4.2 Tag Splitting and Mapping

The system has to determine when and for how long tags mainly appear in the document collection to assign tags to location boxes. However, there can be multiple distinct time spans. For instance, some tag can be mainly mentioned in the beginning of the month as well as in the end of the month, but rarely in-between. Furthermore, two tags can be strongly connected on a particular day, but one of it is mentioned similarly often throughout the week as well. These are cases in which the same tags would need to be placed at distinct locations to reflect the semantics adequately. For this purpose, the concept of tag splitting is introduced.

Figure 4.6 shows an example of tag splitting in the resulting visualization. Here, the term *'south carolina'* often appears around February 26th, but also on the following weekend when it was announced that Joe Biden had won the democratic primary in South Carolina. Splitting the tag allows to assign each tag to one of the respective peaks. This avoids a centered, more misleading placement between those two peaks.

To determine in which time ranges each tag is strongly mentioned, we first count the number of occurrences per day for each tag in the relationship analysis phase. Then, we extract possible time spans by finding contiguous regions in the corresponding occurrences histogram where each count is over a threshold of 70% of the maximum value or is similarly high as a neighboring day that is above the threshold. The resulting one or more time spans are added to the candidate set of spans for this tag. Each time span is defined by a *start day* and *duration* which can be mapped to the corresponding location box.

We also want to collect time spans in which one tag often occurs nearby another tag. That is, even if the occurrence counts of the particular term would not exhibit a pattern of separate peaks, this could still be the case if combined with another tag. For instance, *'world cup'* may appear consistently throughout the week, but *'england'* and *'world cup'* may often appear closely together at one particular day. In this case, we would add this day to the candidate set of time spans for the tag *'world cup'*. To realize this, we compute for each tag i the sum of the order-aware distance weights per day where this tag is involved, that is

$$\sum_j w_{ij}^d + w_{ji}^d$$

where d stands for the respective day. Here, w_{ij} is the weight in cases where tag i appeared *before* tag j, and w_{ji} where tag i appeared *after* tag j. We extract again contiguous regions in the resulting histogram that indicate time spans in which this tag strongly correlates with at least one other tag, and add them to the candidate set of spans for this tag.

This can result in many similar time ranges, for instance, two week-long spans that are just shifted by one day. We need to find a suitable compromise between having many duplicate tags that would lead to visual clutter and too few tags that would lead to misleading layouts. For each span candidate, the respective accumulated count of the underlying histogram (i.e., how many occurrences in the data this time span covers) indicates the significance of the span for its tag. We only select those candidates that are similarly significant as the strongest candidate and sufficiently distinct to the remaining time ranges.

If there is only one time span left, the tag is assigned to the corresponding location box. Otherwise, we split the tag so that each time span can be assigned to one split tag. The document counts of the resulting split tags (for the tag size) are distributed according to the relative significance of the corresponding time range (i.e., how many occurrences it covers). The related order-aware distance weights w_{xy} stay the same.

4.4.3 Particle Swarm Optimization

The system applies Particle Swarm Optimization (PSO) (Chopard and Tomassini [2018]; Shi and Eberhart [2002]) to find a reasonable local minimum of the objective function (4.1). PSO does not use the gradient and the objective function can therefore also include non-differentiable features. Section 2.1.1 provides a more thorough introduction to PSO.

Let A be the total area of all tags combined. To set the working image plane dimensions (p_w, p_h), we define that the area of the image plane should be $4A$ and enforce an aspect ratio of 16:9. The exact size is not important, but the smaller the area, the more dense the resulting layout, with less room for the context- and word order-objectives.

The parameters of the objective function are the positions (p_i^x, p_i^y) on the image plane for each tag i. Thus, the parameter dimension is $2N$ if one wants to place N tags on the plane. The goal is to apply PSO to find parameters that yield a reasonable minimum of the objective function. Technically, the parameters are the positions encoded as fractions relative to the plane width and height, respectively. Before calculating the value of the function, however, we define rectangles that represent the tags and compute their position on the plane.

4.4.4 Objective Function Components

Location Force

Let (c_i^x, c_i^y) be the center of the rectangle representing tag i, (l_i^x, l_i^y) the center of the associated location box and (α_x, α_y) the horizontal and vertical distance of the tag's center to the box boundary if the tag is currently placed outside the box (otherwise 0). The following term pushes tags to be placed inside their associated location box:

$$d_i^l = \sqrt{(|l_i^x - c_i^x| + \alpha_x(1 + \alpha_x))^2 + \left(|l_i^y - c_i^y| + \alpha_y(1 + \alpha_y)\right)^2}$$

$$\text{location} = \frac{1}{\sqrt{A}} \sum_i^N d_i^l$$

If the tag is placed inside the boundary of its location box, the regular distance between the center of the tag and the center of the location box is penalized, pushing tags slightly towards the center of the box. If it is outside, however, we strongly increase the weight of the penalty with an additional squared term based on the distance to the boundary of the box.

The square root of the total area of all tags combined ($\sqrt{A}$) is used for normalization. If tag splitting is disabled and every tag is assigned to the same 'location box' spanning the entire image plane, then this method generates classic tag cloud layouts in which tags are densely placed in the center of the image.

Collision avoidance

Let I, J be the area of tag i and j, respectively. We compare the rectangles representing tags with each other and compute the intersecting area $|I \cap J|$. The total intersecting area makes up the collision avoidance component, again normalized by $\sqrt{A}$:

$$\text{collision} = \frac{1}{\sqrt{A}} \sum_{i}^{N} \sum_{j=i+1}^{N} |I \cap J|$$

Proximity

Tags that are related to each other should also be close to each other, that is, the distance between any two tags i and j should be minimized, weighted by their relatedness r_{ij}. Section 4.3.2 explains how the order-aware distance weights w_{ij} are calculated using the inverse distance between pairs of tags in the source data. These weights influence the relatedness score:

$$r_{ij} = \frac{1}{Z}(w_{ij} + w_{ji}) \max(a_i, a_j) \Phi_{ij}$$

$$a_x = \log \mathrm{IDF}(t_x) \quad Z = \frac{1}{N} \sum_{i}^{N} \max_{j}(w_{ij} + w_{ji})$$

We normalize the order-aware distance weights with Z to retrieve relative values between 0 and 1. Phrases that generally appear often in documents are more likely to occur nearby just by chance. Hence, we further apply a correction factor a_x which is the logarithm of the inverse document frequency of the most frequent term. This restricts the influence of high-frequency tags, but also prevents a strong emphasis of low-frequency outliers, which is similar to the idea of TF-IDF, where the influence of terms occurring in many documents is diminished. As a result, the relatedness between two tags is high if they appear unusually often close to each other in the documents.

The previously described tag splitting may result in multiple occurrences of the same two tags (content-wise) at different time ranges. The connection between each pair as expressed by the relatedness should also depend on the overlap of the time spans they have been associated to. For instance, if *'australian open'* and *'andy murray'* are assigned to a location box on the left and there are two additional tags *'australian open'* and *'djokovic'* which are assigned to the right, then the first *'australian open'* should mainly be related to *'andy murray'* and not to *'djokovic'* on the right. Therefore, we weight the order-aware distance-weights of a pairing (calculated on the whole time range) with Φ_{ij} that represents the share of the *global* relatedness for this particular pair based on the overlap of their associated time spans.

Let l_i^c be the column and l_i^r the row of location box assigned to tag i. The proximity component of the objective function penalizes large distances between tags that have a high relatedness:

$$\hat{d}_{ij}^o = \frac{d_{ij}^o}{\sqrt{A}} \qquad \phi_{ij} = \frac{1}{1 + (|l_i^c - l_j^c| + |l_i^r - l_j^r|)\frac{5}{n_d}}$$

$$\text{proximity} = \sum_{i}^{N} \sum_{j=i+1}^{N} \hat{d}_{ij}^o (1 + \hat{d}_{ij}^o) r_{ij} \phi_{ij}^2$$

Rather than calculating the distance between the center coordinates of the rectangles, we instead use the outer distance d^o that is defined as the closest distance between two points on the border of each rectangle. This ensures that we do not penalize horizontal stacking, that is, tags placed next to each other horizontally have the same outer distance of zero like tags that are stacked on top of each other.

If two tags are assigned to different location boxes that are far away, it is nearly impossible to place them nearby. The proximity force is reduced in these cases by multiplying with the square of ϕ_{ij} which is the inverse difference between row and column indexes of the respective location boxes. The term $5/n_d$ is just for normalization to retrieve values independent from the total number of days.

Repelling Force

On the one hand, we would like to have a compact visualization. On the other hand, we would also like to slightly separate tags if the underlying data suggests that these tags are not related to each other. Whereas the previous proximity component rewards close distances of related tags, the following repelling force encourages tags to be placed slightly apart. In combination with the proximity term, this should lead to better visual clusters of related concepts. Let d_{ij} be the distance between the center points of the tags t_i and t_j. The repelling component is then defined as:

$$\text{repelling} = \sum_{i}^{N} \sum_{j=i+1}^{N} \frac{1}{1 + \left(d_{ij}\right)^{1.5} \sqrt{A}}$$

The intuition behind the exponent $1.5 > 1$ is to let this force quickly diminish with increasing distance.

Word Order

In some cases, the data implies a certain ordering of tags, for instance, if *'tree'* and *'christmas'* appear next to each other, *'tree'* comes after *'christmas'*. The layout should preserve these word order characteristics in the visualization as best as possible to improve the readability and support sensemaking tasks. As previously explained, if the order-aware distance weight w_{xy} is much greater than w_{yx}, then tag x often appears before tag y when they occur nearby in the documents.

We define the word order o_{ij} which ranges from zero (tag i is right to tag j) to one (tag i is left to tag j), and the strength of the ordering γ_{ij} to express how certain the system is that the data implies an ordering between tag i and j:

$$o_{ij} = \frac{w_{ij}}{w_{ij} + w_{ji}}$$

$$\gamma_{ij} = \left(0.5 - \min(o_{ij}, 1 - o_{ij})\right) \left(\frac{10}{z} \cdot |w_{ij} - w_{ji}|\right)^{0.3}$$

The first part of γ is zero if there is no implied word order, that is, tag i appears equally often before and after tag j. It reaches its maximum at 0.5 if all occurrences are in the same order. The second part expresses how strong the evidence is and uses the absolute difference between the order-aware distance weights.

The intuition behind having an exponent < 1 is that these differences do not follow a uniform distribution. A linear relationship would underestimate the evidence except for the pair with the highest difference. The normalization constant z is the maximum number of tag pair occurrences and, therefore, an upper bound for w.

In combination, γ is high when tag i and j often appear in close proximity with one specific order for the most part. The normalization constant 10 and the exponent 0.3 were empirically determined such that the resulting forces are reasonable across different data sets, but they can be changed to shift the emphasis. For instance, a higher exponent increases the required evidence from the data, resulting in less tag pairs that are considered to have a significant word order.

The function penalizes the horizontal distance if the tag is on the wrong side according to the word order, weighted by the strength of the ordering:

$$x_{ij} = \begin{cases} \max\left(-\mathrm{width}(t_i), p_i^x - p_j^x\right) & \text{if } o_{ij} >= 0.5 \\ \max\left(-\mathrm{width}(t_j), p_j^x - p_i^x\right) & \text{otherwise} \end{cases}$$

$$\mathrm{wordOrder} = \frac{1}{\sqrt{A}} \sum_{i}^{N} \sum_{j=i+1}^{N} x_{ij} \gamma_{ij} r_{ij} \phi_{ij}$$

The lowest and best value for x_{ij} is reached if the tag that should appear first is completely placed before the other tag. The worst value is only limited by the image plane boundaries. We cap values at the optimal value $-\mathrm{width}_{i/j}$, because we do not want to encourage tags to be pushed too far away again. Similar to the proximity component, we take the relatedness and the location box distances of the tags (if applicable) into account as well.

4.5 Evaluation

The first part of this section presents a use case scenario and shows a comparison between the PyramidTags layout and more traditional word cloud layouts to illustrate the usefulness of the approach. Then, the results of a quantitative evaluation on a benchmark data set are discussed. The section finally reports on qualitative feedback that was gathered from two domain experts.

4.5.1 Use Case Scenarios

Figure 4.7 presents the resulting visualization of about 150,000 English news articles mainly from the US and the UK during the whole month of August 2020. From this static visualization (i.e., before any interaction is performed) the analyst can already form several hypotheses. For instance, the tags at the

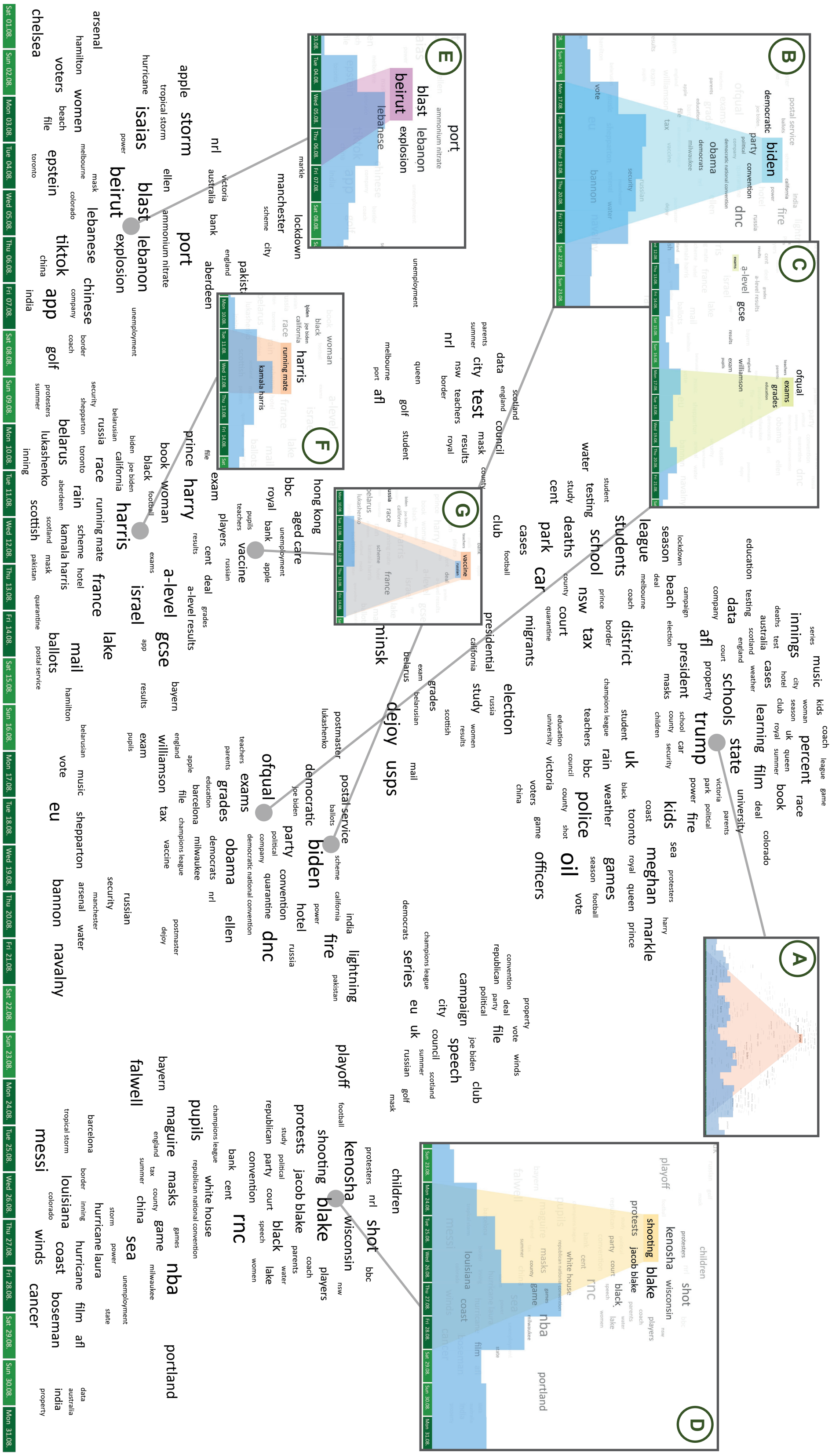


Figure 4.7 — PyramidTags of about 150,000 news articles published in August 2020 with 200 distinct tags. The cutouts A–G show how the visualization adapts when the user hovers over the respective terms.

very top indicate that news regarding Donald Trump (*'president ... trump'*) and baseball (*'innings', 'afl'*) dominated the entire month. This also applies to articles about the Royals (*'meghan ... markle, queen, harry'*), with a slightly stronger emphasis towards later days in August.

The bigger tags that are closer to the timeline at the bottom suggest that something has happened in Beirut in the first week of the month, that Kamala Harris was often on the news during the second week, and that the Democratic as well as the Republican National Convention was often the subject of reporting in the third and fourth week, respectively.

Moving the mouse over tags helps to further confirm or reject hypotheses that one might have gained by looking at the tags. For instance, the terms *'shooting'*, *'protests'*, and *'blake'* on the right of the visualization seem to be related. After selecting the tag *'shooting'* (D), it becomes clear that this is indeed the case. The retrieved articles report about ongoing protests and unrest regarding the shooting of Jacob Blake. However, hovering over *'biden'* (B) reveals that the term *'scheme'* right above it does not seem to be related.

Summarizing 150,000 articles with just 200 tags may seem to be challenging at first, but the number of possible distinct combinations is huge, even even with just two or three tags. For instance, there are many (> 3,000) posts that contain *'vaccine'* or *'russian'*, but after selecting *both* tags (G), the bar chart and the search results show that there are only 311 articles (mainly published on August 11th and 12th) that contain both terms, talking about the announcement of the Russian government that they had developed and approved a Covid-19 vaccine. In the presented use case, the first 100 search results after selecting up to two tags cover about three quarters of all articles.

The objective function is composed of different components that can also be activated individually to generate more traditional tag cloud layouts. Figure 4.8 shows two examples based on about 160,000 articles published in January 2020. The first version at the top shows a classical tag cloud, which was generated using only the collision avoidance component and one global location box spanning the entire image plane. It shows that many news articles talked about Brexit, the Iran, or the NFL, but the visual encoding offers only a limited and shallow insight into the data set. The second version at the bottom is more context-aware and word order-aware, which helps to make better sense of the content. For instance, several tags related to the Royal familiy are clustered together, and the tags around *'iraq'* and *'iran'* indicate that something has happened in the Middle East.

Figure 4.9 depicts the same data set but with the full PyramidTags layout. Compared to the previous versions, the example shows that the triangular

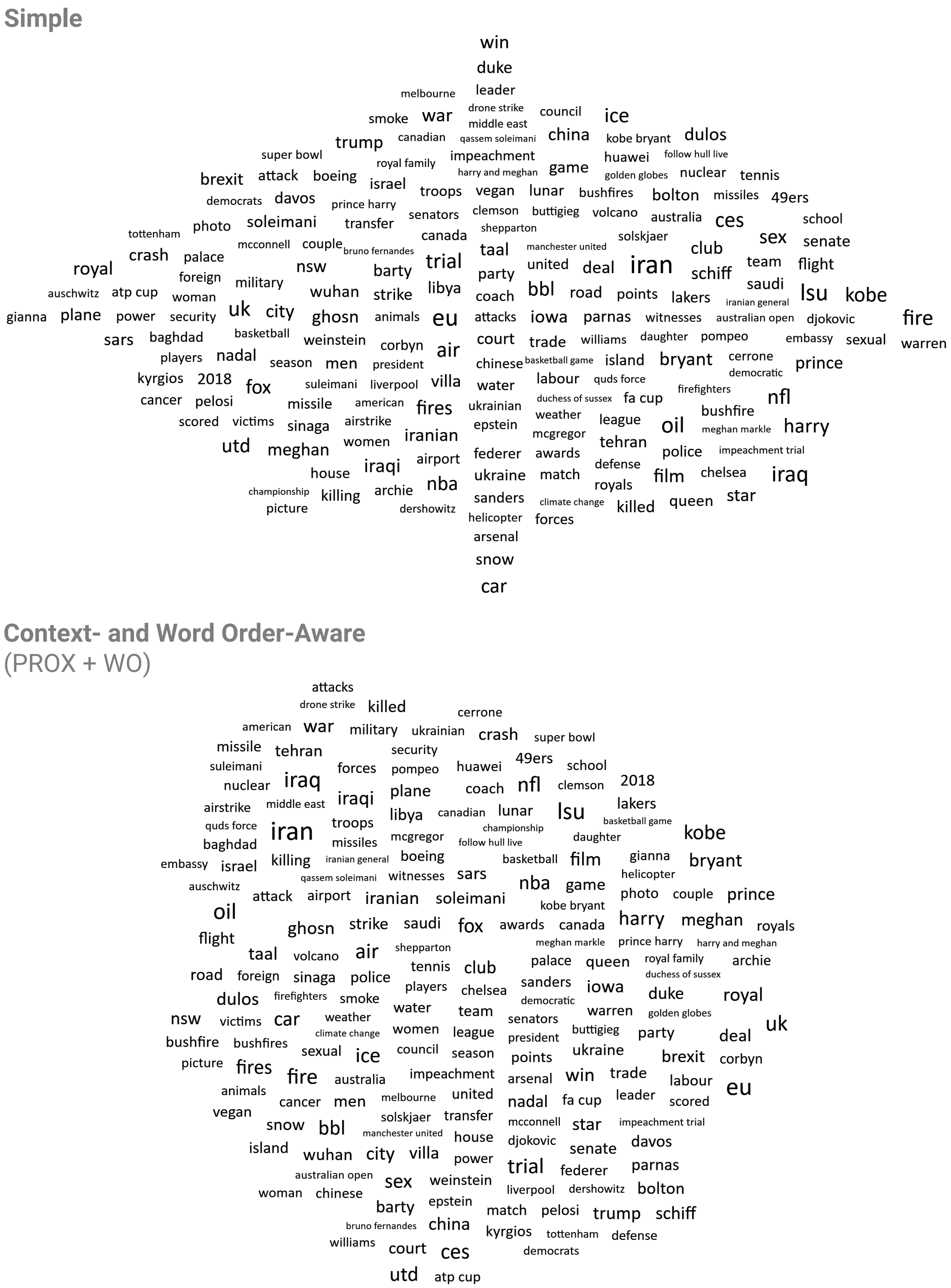

Simple
win
duke
melbourne leader
drone strike council ice
smoke war middle east
trump canadian qassem soleimani china kobe bryant dulos
super bowl impeachment huawei follow hull live
royal family game
brexit attack boeing israel harry and meghan golden globes nuclear tennis
troops vegan lunar bushfires bolton missiles 49ers
democrats davos prince harry
photo soleimani transfer senators clemson buttigieg volcano australia ces school
tottenham canada shepparton solskjaer sex senate
crash palace mcconnell couple bruno fernandes taal manchester united club
royal trial united deal iran team flight
foreign military nsw barty party schiff saudi
auschwitz atp cup woman wuhan libya bbl road points lakers iranian general lsu kobe
strike coach
gianna plane power security uk city ghosn animals eu attacks iowa parnas witnesses australian open djokovic fire
sars baghdad basketball weinstein court trade williams daughter pompeo embassy sexual warren
players nadal corbyn air
season men president chinese basketball game island bryant cerrone prince
kyrgios 2018 democratic
fox suleimani liverpool villa water labour quds force
cancer pelosi missile american ukrainian duchess of sussex fa cup firefighters nfl
fires weather league bushfire
scored victims sinaga airstrike epstein oil meghan markle harry
iranian mcgregor
utd meghan women tehran police impeachment trial
airport federer awards
iraqi defense
house ukraine match film chelsea iraq
championship killing archie nba royals
picture sanders climate change queen star
dershowitz killed
helicopter forces
arsenal
snow
car
Context- and Word Order-Aware
(PROX + WO)
attacks
drone strike killed
cerrone
american war military ukrainian crash super bowl
missile tehran security
suleimani forces pompeo huawei 49ers school
nuclear iraq coach nfl clemson 2018
iraqi plane
airstrike middle east lakers
libya canadian lunar lsu
quds force troops championship basketball game
iran kobe
baghdad missiles mcgregor follow hull live daughter
killing iranian general boeing
embassy israel basketball film gianna bryant
qassem soleimani witnesses sars helicopter
auschwitz nba game
attack airport iranian soleimani photo couple prince
kobe bryant
oil harry meghan
ghosn strike saudi fox awards canada royals
flight meghan markle prince harry harry and meghan
taal volcano air shepparton royal family archie
palace queen
tennis club duchess of sussex
road foreign sinaga police
players chelsea sanders iowa duke
dulos firefighters smoke royal
water team democratic
nsw victims car weather senators warren golden globes
climate change women league president buttigieg party uk
bushfire bushfires deal
sexual ice council season points ukraine
picture brexit corbyn
fires impeachment trade
fire australia arsenal win labour
animals melbourne united leader eu
cancer men nadal fa cup scored
vegan solskjaer transfer
mcconnell star impeachment trial
snow bbl manchester united house
djokovic senate davos
island
wuhan city villa power trial federer parnas
australian open
sex weinstein
woman chinese liverpool dershowitz bolton
barty epstein
match pelosi trump schiff
bruno fernandes china kyrgios
tottenham defense
williams court ces
democrats
utd atp cup

Figure 4.8 — Two traditional tag cloud layouts (one global location box) with 200 distinct tags based on about 160,000 articles published in January 2020, generated using only a subset of objectives. **Top:** simple tag cloud with only the collision and location force components enabled. **Bottom:** context- and word order-aware tag map.

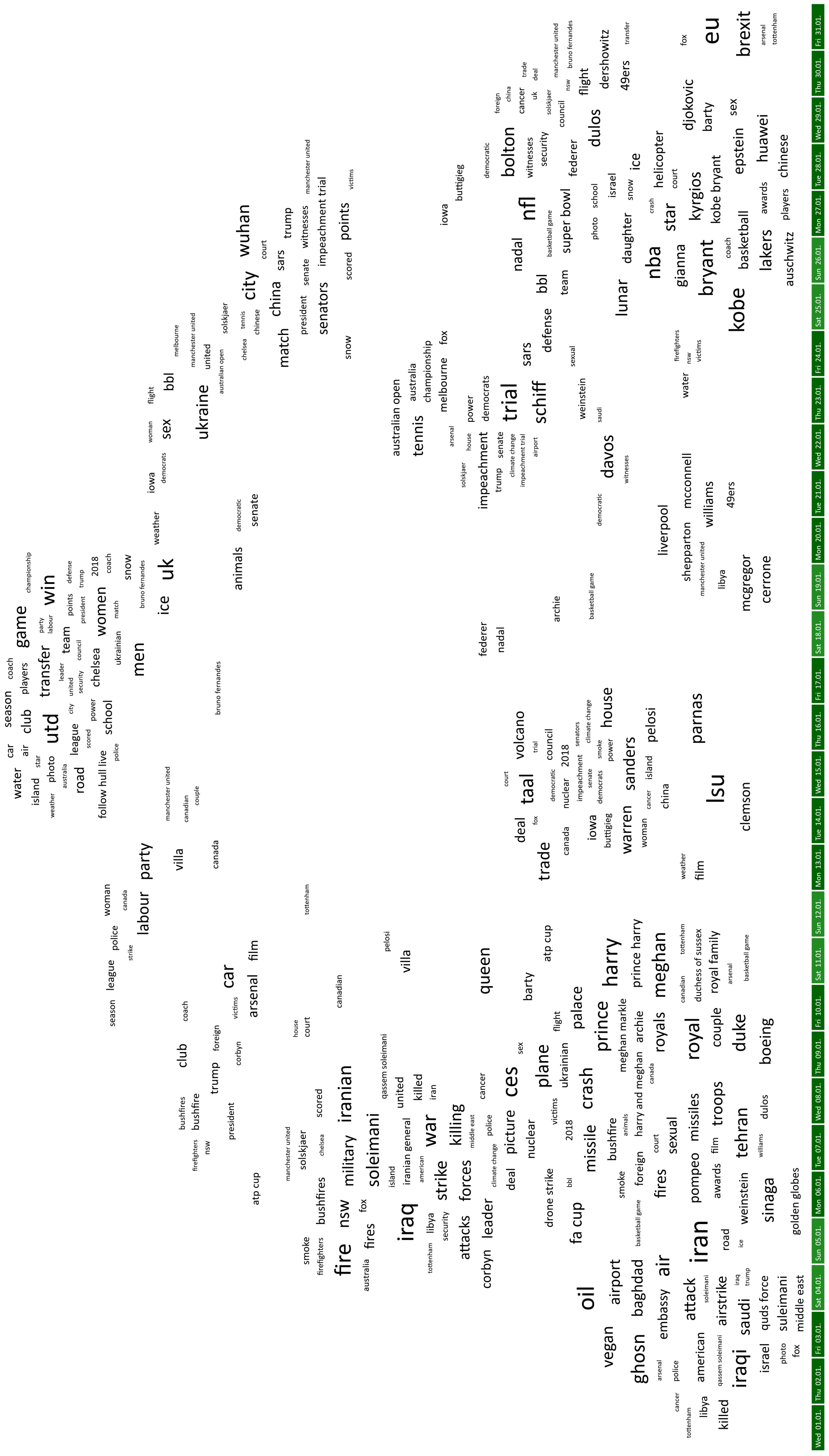


Figure 4.9 — Full PyramidTags layout with 200 distinct tags based on about 160,000 articles published in January 2020.

layout helps to understand the development of certain topics over time, for instance, that many articles talked about Wuhan and Sars in the second half of the month, and that Kobe Bryant's helicopter crash happened at the end of January.

4.5.2 Benchmarks

It is difficult to evaluate tag cloud layouts and similar approaches dealing with text spatializations since task-driven studies often fail to cover non-specific information seeking needs. Such needs are difficult to enfore and stage artificially. To tackle these challenges, the approach was evaluated quantitatively by utilizing meta-data from collected news articles. This benchmark data set comprises about 60,000 articles, annotated with a list of associated *topics*, from January to December 2019 from the British newspaper `The Guardian`. Typically, this list contains both broad categories such as *US News* and also more specific ones, for instance, *Immigration Policy*. These annotations are used to test the context-awareness of the approach.

Two configurations were evaluated: a time span of two weeks with 100 tags and one month with 200 tags. The test script trained several different visualization types by successively activating components of the objective function (Equation 4.1), that is, setting the respective $\lambda_x \neq 0$. *Simple* is the regular multi-word tag cloud (only collision component activated, one location box). *PROX* or *WO* means that the `proximity` or `wordOrder` component is enabled, respectively. *Pyramid* indicates types with the triangular layout, different location boxes and tag splitting. Hence, the last row represents the full PyramidTags variant with all $\lambda_x \neq 0$.

The test computed the following evaluation criteria for the generated output:

Density: Average number of tags in the neighborhood of any tag. The neighborhood of a tag consists of all tags that are located within an outer distance (shortest distance between the bounding boxes) of the height of the respective tag. Choosing a slightly higher threshold increases the number of tags in the neighborhood, but the relations of the benchmark values between different variants stay largely the same.

Context-Overlap (Context): Benchmark for assessing how thematically related neighboring tags are. For each tag, the script fetches all articles that contain the tag. Each article has a list of categories provided by the newspaper, which is used to build a *topic vector* with the occurrence count of each topic, representing the thematic landscape of the respective tag. *Context* is the average cosine similarity between the topic vectors of the center and a neighboring tag. A higher value indicates that nearby tags are more likely to be related thematically. To make these values more interpretable, the script further calculates the probability that a neighboring topic vector has a cosine similarity ≥ 0.7 (*Context7*), that is, how likely it is that a neighboring tag is thematically very related (assuming the respective threshold).

Word-Order (WO*x*): For each tag set, the script creates a ranking of pairs according to the evidence that they appear in a certain order as explained in Section 4.4.4. To evaluate how well different layout strategies perform in preserving the word order, it calculates the fractions of the top 5 (*WO5*), 10 (*WO10*) and 50 (*W50*) pairs appearing in the correct order.

Date-Awareness (DateCos): For each tag, the script builds a date vector comprised of the number of articles per day. *DateCos* denotes the average cosine similarity between date vectors of a tag and one of its neighbors. A higher value indicates that neighboring tags have a higher similarity regarding the temporal evolution of related articles.

4.5.3 Results

Similar to the training of neural networks, the output may differ due to the random initializations. Thus, this section reports the average values of three runs to mitigate the effect of outliers. Table 4.1 shows the results of the benchmarks for the configuration of 100 tags and a time span of two week, and Table 4.2 for the configuration of 200 tags and one month.

All visualization types with activated proximity component show a notable improvement in the Context-Overlap score. The triangular layout resulted in an increase of the cosine similarity between date vectors of neighboring tags compared to the simple layout. Interestingly, the benchmarks show that it is also more context-aware even without enabling the proximity component, indicating that tags that mainly appear during similar time ranges are also more likely to be related.

The word order of the most important pairs is mostly preserved in all variants that use the `wordOrder` component. Conversely, the word order probabilities of the remaining variants are on par with the expected random odds. However, it can be seen that there is a trade-off between context-awareness and word-order preservation.

As expected, the *Pyramid* layouts have a lower density compared to the other variants. Nevertheless, the PyramidTags approach stands out for its unique ability to express temporal relationships while at the same time preserving the word order at least as good as the *PROX+WO* variant.

Table 4.1 — Benchmark results for different layouts with 100 tags and two weeks (higher is better). *WO*, *PROX* and *Pyramid* denote whether the respective `wordOrder`, `proximity` and `location` components of the objective function are activated. The last row represents the full PyramidTags layout.

	Density	Context	Context7	WO5	WO10	WO50	DateCos
Simple	7.7	0.33	0.17	0.52	0.52	0.50	0.76
WO	7.1	0.35	0.19	0.99	0.99	0.94	0.76
PROX	7.9	0.50	0.37	0.52	0.51	0.52	0.78
PROX + WO	7.8	0.49	0.36	0.73	0.71	0.64	0.78
Pyramid	4.1	0.40	0.26	0.55	0.55	0.52	0.80
Pyramid + WO	4.1	0.40	0.26	0.84	0.84	0.69	0.80
Pyramid + PROX	4.9	0.45	0.31	0.53	0.56	0.52	0.82
Pyramid + PROX + WO	4.9	0.45	0.31	0.77	0.76	0.60	0.82
Pyramid + PROX + WO + REP	4.6	0.45	0.31	0.78	0.74	0.60	0.82

Table 4.2 — Benchmark results for different layouts with 200 tags and one month (higher is better). *WO*, *PROX* and *Pyramid* denote whether the respective `wordOrder`, `proximity` and `location` components of the objective function are activated. The last row represents the full PyramidTags layout.

	Density	Context	Context7	WO5	WO10	WO50	DateCos
Simple	7.8	0.38	0.17	0.56	0.55	0.54	0.77
WO	7.4	0.40	0.20	0.99	0.99	0.97	0.78
PROX	8.1	0.55	0.40	0.51	0.51	0.51	0.79
PROX + WO	8.1	0.54	0.38	0.70	0.67	0.66	0.79
Pyramid	4.7	0.43	0.25	0.53	0.52	0.55	0.80
Pyramid + WO	4.6	0.43	0.24	0.88	0.82	0.76	0.79
Pyramid + PROX	5.3	0.47	0.29	0.54	0.54	0.54	0.82
Pyramid + PROX + WO	5.3	0.47	0.29	0.84	0.79	0.70	0.82
Pyramid + PROX + WO + REP	4.9	0.47	0.30	0.84	0.77	0.69	0.82

These benchmarks reveal that the proposed approach is clearly more context-, word order- and date-aware than traditional tag clouds. Furthermore, they show that the proposed placement strategy to visualize the temporal evolution of tags structures the data in a meaningful way.

In addition, the visual comparison in Figure 4.8 reveals that the *PROX+WO* approach offers semantically clustered tag clouds while also better preserving the word order compared to a random layout, but with the same high density and no increase in screen space, making it well-suited for data sets without timestamps or in which the temporal evolution only plays a marginal role in analyzing the data.

4.5.4 Qualitative Feedback

In addition, PyramidTags was presented individually to two visualization experts working in a company that deals with social media monitoring. First, they had to inspect the visualization of one month without any introduction and tell the instructor what they thought the data set was about, how they think the layout is supposed to work, and which insights they gained. After explaining the system and which kind of interactions it supports, the instructor then asked them to further explore the data set and try to find out more about specific themes they are interested in.

Both experts stated that the triangular layout is intuitive, however, one expert initially thought that the pyramid represented a hierarchical representation of topics before the explanation of the system. They found it helpful that many tags were placed in their reading direction and that the dots indicate the assumed word order. Both quickly noticed several clusters of tags and hypothesized about related stories and their temporal evolution. They rated the interface as responsive, with no lag while interacting with it.

One expert was wondering why there were duplicate tags, but found it useful after reason behind it was revealed. They noted that, in some cases, nearby tags wrongly appeared to belong together, and suggested some sort of coloring or a visual border to separate unrelated tags, if possible at all.

One expert was slightly irritated that clicking tags was an additive operation, in contrast to other applications that usually require the user to press a modifier key. While different colors for different tags were regarded necessary for distinction, one interviewee proposed to always start with the same color for the first selected tag and change the hue for every subsequently selected tag.

One expert would have found it useful to highlight tags based on a selected date range at the bottom. Furthermore, they stated that also offering a negative

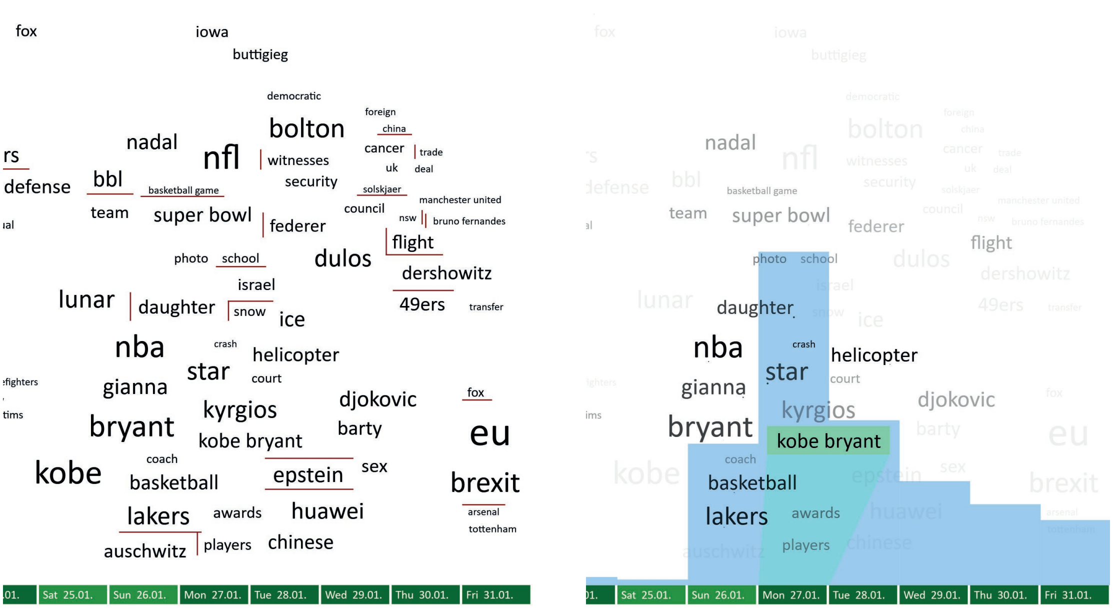


Figure 4.10 — Tags that are close to each other are not necessarily related. **Right:** hovering over *'kobe bryant'* reveals that it does not relate to the nearby term *'epstein'*. **Left:** thin red lines visually separate tags that are placed next to each other but are not related.

selection, that is, selecting tags that are not relevant, would improve the utility. Both proposed to implement a filtering mechanism that would allow analysts to explicitly search for keywords, resulting in an updated visualization solely based on the filtered documents.

They thought that the presented search results were relevant to the selection they performed and liked the fact that documents open in a new tab instead a new window. One suggestion was to highlight occurrences of selected tags in the retrieved document. Both regarded the overall system as helpful to explore large time-stamped collections. One expert was particularly impressed that the approach made it easy to find specific topics with just two or three clicks, even though, at first glance, the visualization had not looked like it would really represent that many documents.

4.6 Discussion

The visual metaphor of a tag map has the advantage that the spatial positioning of the summarizing tags can convey semantic relationships while avoiding predefined topics. The proposed spatialization technique enables analysts to not only track individual tags over time, but also to quickly see at which date range a group of possibly related tags is most present in the collection. This also means that it aims at summarizing time-stamped documents whose associated date is relevant to the analysis process, which is often the case with news and social media corpora, for instance.

The relatedness between tags is induced by their distance patterns in the underlying data, resulting in a more fine-grained concept of relationship compared to a range of previous approaches that define semantic similarity based on whether tags appear in the same sentence.

The interaction techniques support analysts to select appropriate tags of interest and retrieve related documents. If many tags are selected at the same time, the (optional) trapezoids visualizing the respective time ranges become less useful due to overlap. However, it rarely happens that this many tags have to be selected, because every new item drastically reduces the search space.

A disadvantage of the approach is the increase of whitespace compared to dense tag cloud layouts. This increase can limit the usability and utility in certain cases, for instance on mobile devices. However, whitespace does not necessarily mean wasted space, because the lack of content can also express information such as less activity in certain date ranges, or less semantic coherence. Furthermore, the space on the upper left and right can be used to place the search results window and document viewer.

Tags that are placed close to each other do not necessarily relate to each other. Additionally, in some cases viewers might 'read' a certain phrase which makes sense grammatically, but is actually not backed by the document collection. These are common disadvantages if complex non-linear relationships are projected onto flat visualizations. Here, users can interactively debunk such false friends, for example by hovering over tags to reveal relationships in detail. One of the experts raised the idea to draw borders between unrelated tags, which would help to avoid misleading hypotheses. An exemplary implementation is shown in Figure 4.10. Here, the thin red lines make immediately clear that the term *'epstein'* has nothing to do with *'kobe bryant'*, even though both tags are placed close to each other. However, this does not solve the issue completely as these borders also increase the visual separation to potentially related tags that are placed further away.

In contrast to many related approaches, the method was applied to large, real-world data sets in order to prove its scalability. On a modern 6-core CPU with a parallel implementation, it takes about 15 seconds to process 10,000 documents extracting 100 tags, analyzing their relationship and generating an index for document retrieval. Processing 500,000 documents extracting 250 tags lasts 10-20 minutes. Generating the final visualization additionally takes several minutes. Once the visualization is generated, users can interact with it and retrieve documents instantly.

Aiming for several objectives at the same time is challenging. Nevertheless, the quantitative evaluation showed that the approach reliably produces semantically clustered layouts which also convey temporal patterns *and* preserve the word order for the most important pairs. In combination with rich interactions, the system enables analysts to explore and gain insights into large document collections.

Chapter 5

Dynamic Document Clustering

As outlined in Section 2.1.2, clustering algorithms facilitate the analysis of large data sets, particularly if they comprise unstructured data such as textual documents. The automatic structuring of data sets into groups of similar items helps analysts with managing and organizing large collections. In addition to the resulting grouping, the clusters themselves may also reveal interesting characteristics of the underlying data set. In the context of the AIX framework that was presented in Section 2.5, this refers to visualizing and understanding the model itself and not only its outputs, the groups of documents. The defining characteristics of each cluster can make relationships in the data set visible because they essentially represent aggregations of similar items. For instance, a given clustering of documents might reveal which terms relate to a particular topic and are often used in a set of similar documents, even though the actual terms may rarely co-occur in the same document. Not every clustering algorithm allows these inspections, though. For instance, if we populate groups by subsequently finding the nearest neighbors of all elements, we get a clustering, but no characterizing definition of each group without further processing.

Dhillon and Modha [2001] proposed Spherical k-Means for the clustering of documents that have high-dimensional ($\gg 1000$) but very sparse vector representations. It is based on k-Means (Lloyd [1982]) and replaces the Euclidean with the cosine distance to improve the performance on documents. Section 2.1.2 gives a more detailed overview of the algorithm. Spherical k-Means performs competitively on textual data sets (Lelu and Cadot [2021]), is computationally efficient if k is sufficiently small, and, most importantly, allows for a visual inspection of the resulting centroids that define the clusters, given a word-based vector representation of the documents. It is therefore a suitable choice for making structures and relationships in large collections visually apparent.

Table 5.1 shows an example how clustering helps to organize and structure the content of text collections. It is based on the *20 Newsgroups* data set[1] containing about 20,000 posts across 20 different newsgroups. The left column shows the most important terms for each newsgroup, and the right column for each extracted cluster after the Spherical k-Means algorithm was applied to all posts (the clusters have been rearranged for an easier comparison). Many extracted clusters have striking similarities to one of the theme-specific newsgroups.

However, k-Means generally does not scale well with the number of clusters, making fine-grained analyses of large document collections infeasible, particularly in interactive settings. Furthermore, the algorithm does not support the dynamic clustering of streaming data. The discussion in Section 2.1.2 shows that there is generally a lack of algorithms that enable an efficient dynamic clustering for the visual analysis of streaming documents. Hence, this chapter proposes an efficient and interpretable dynamic clustering algorithm that is based on an accelerated version of Spherical k-Means. The first part presents and evaluates the two strategies for accelerating the Spherical k-Means algorithm, and the second part details the proposed changes for making it dynamic.

5.1 Efficient Spherical k-Means

On each iteration of (Spherical) k-Means, one has to find the closest cluster centroid for every data item, which results in $\mathcal{O}(kN)$ comparisons where one has to compute the distance between two vectors. The linear dependency of k on the time complexity can lead to prohibitively large running times on larger data sets with $k \gg 10$.

Several approaches have been developed to increase the computational efficiency of k-Means, but they cannot be applied to *Spherical* k-Means on sparse document representations. Elkan [2003] applied the triangle inequality to reduce the number of distance calculations that have to be performed, but the triangle inequality does not hold for the cosine distance function. Data structures for efficient nearest neighbor searches that are based on, for instance, k-d trees (Bentley [1975]), coordinate-pruning (Teflioudi and Gemulla [2016]), or product quantization codes (Johnson et al. [2021]), generally assume moderately-sized, dense input vectors, or are based on the triangle inequality. Numerous efficient

[1] http://qwone.com/ jason/20Newsgroups/

Table 5.1 — Clustering helps to visually structure unlabeled text collections. **Left:** Most important terms for each newsgroup in the 20 Newsgroups data set. **Right:** Most important terms for each cluster after applying Spherical *k*-Means to all posts of the 20 Newsgroups data set. The clusters have been rearranged based on their similarity to a newsgroup.

Newsgroup	**Cluster**
1: god, morality, atheists, objective, atheism	1: objective, morality, moral, abortion, o'dwyer
2: graphics, image, 3d, images, files	2: 00, graphics, $, 3d, xv
3: windows, dos, file, microsoft, files	3: windows, dos, mouse, drivers, file
4: drive, ide, scsi, card, controller	4: drive, card, scsi, mac, ide
5: mac, apple, monitor, centris, drive	5: apple, kent, duo, sandvik@newton, alink
6: window, motif, server, xterm, x11r5	6: window, program, file, image, files
7: sale, 00, offer, shipping, forsale	7: sale, offer, condition, shipping, cd
8: car, cars, engine, oil, ford	8: car, cars, engine, oil, radar
9: bike, dod, ride, bmw, riding	9: simms, georgia, simm, v6, uga
10: baseball, game, team, games, players	10: baseball, game, games, players, year
11: game, hockey, team, nhl, espn	11: game, hockey, team, nhl, espn
12: clipper, key, encryption, chip, keys	12: clipper, key, encryption, chip, bike
13: circuit, copy, battery, radio, power	13: power, battery, circuit, test, voltage
14: msg, doctor, medical, disease, photography	14: msg, medical, doctor, disease, drugs
15: space, nasa, moon, shuttle, orbit	15: space, nasa, moon, shuttle, orbit
16: god, jesus, church, christ, christians	16: god, jesus, bible, church, christian
17: gun, fbi, guns, batf, atf	17: gun, guns, people, government, rights
18: israel, israeli, turkish, jews, armenian	18: israel, israeli, jews, turkish, armenian
19: cramer, people, gay, government, clayton	19: fbi, koresh, batf, fire, waco
20: god, jesus, objective, morality, christian	20: theory, evolution, science, creationism, uiuc

online versions of k-Means have been proposed, but these approaches only approximate the k-Means objective.

In this section, an accelerated version of Spherical k-Means is presented that can efficiently cluster large document collections even if $k \gg 10$. The vast majority of the running time is spent on calculating the cosine similarity between an input vector and a cluster centroid. Both proposed strategies aim to reduce the average number of these computations that have to be performed. The first strategy exploits the observation that the number of changing cluster centroids typically decreases after several iterations. The second strategy employs an indexing structure that leverages the sparsity of the input vectors for an efficient (yet non-approximated) retrieval of cluster centroids that maximize the cosine similarity.

5.1.1 Method

This section details the two complementing strategies to accelerate the Spherical k-Means algorithm on sparse document representations.

Non-Changing Clusters (NCC)

Over the course of the iterations, the number of affected data items typically decreases, that means, fewer and fewer data items change their cluster association during the assignment step. As a result, there is an increasing number of cluster centroids that stay the same in later iterations. We can take advantage of this to skip some of the comparisons if the element belongs to one of these clusters.

After recalculating the cluster centroids, we determine the set of centroids C_u that have not changed compared to the previous iteration (within a certain tolerance ϵ to accommodate for rounding errors). During the assignment step, we then check whether the current data item $\mathbf{x}_i$ was previously associated with a cluster in C_u. If this is the case, we only need to calculate the similarity of $\mathbf{x}_i$ to centroids $c_j \notin C_u$ that have actually changed and compare whether we get a higher similarity than with our previous association a_i. We already know from the past iteration that a_i relates to the most similar centroid among C_u since these clusters have not changed in the current iteration.

Dot Product Indexing Structure (INDEX)

In the assignment step, we search for the centroid that minimizes the cosine distance to the current item and, thus, maximizes the dot product with the current item since we operate on unit-length vectors. For sparse input vectors, it could very well be the case that the non-zero entries of a centroid do not overlap with the non-zero entries of the current item, leading to a dot product of 0. To ignore such centroids, we could build an inverse index at the beginning of this step that maps an index to all centroids that have a non-zero value at that position. Given an item, we can then enumerate through all of its non-zero values to retrieve the union of the centroids that share at least one non-zero entry. The dot product with any remaining centroid is zero. This can lead to a measurable speed-up if both the average number of non-zero input vector entries is sufficiently small and the centroids are distinct enough. However, even in sparse settings one cannot generally assume that this is the case.

To improve this indexing structure we can exploit the fact that the input and centroid vectors have length one, and that the dot product with the (possibly updated) centroid based on the previous assignment will most likely be greater than zero.

Lemma 5.1.1. *Given two unit-length vectors* $\mathbf{c} = [c_1, ..., c_n]^\top, \mathbf{x} = [x_1, ..., x_n]^\top \in \mathbb{R}^n$ *and let* $S = \{a_1, ..., a_m\}, a_i \in \mathbb{N}$ *be the set of indexes that correspond to a non-zero value in* $\mathbf{x}$*. That is,* $x_i \neq 0$ *if and only if* $i \in S$*. If* $\mathbf{c} \cdot \mathbf{x} \geq \lambda$ *then it holds that* $\sum_{i=1}^{m} c_{a_i}^2 \geq \lambda^2$*.*

Proof. Let $\mathbf{c} \cdot \mathbf{x} \geq \lambda, \hat{\mathbf{c}} = [c_{a_1}, ..., c_{a_m}]^\top, \hat{\mathbf{x}} = [x_{a_1}, ..., x_{a_m}]^\top$. It follows that $\mathbf{c} \cdot \mathbf{x} = \hat{\mathbf{c}} \cdot \hat{\mathbf{x}}$, because $\hat{\mathbf{x}}$ comprises all non-zero entries of $\mathbf{x}$. We can rewrite the dot product as follows: $\hat{\mathbf{c}} \cdot \hat{\mathbf{x}} = \|\hat{\mathbf{c}}\| \|\hat{\mathbf{x}}\| \cos \hat{\theta}$ where $\hat{\theta}$ is the angle between both vectors. It follows that $\|\hat{\mathbf{c}}\| \cos \hat{\theta} \geq \lambda$, because $\|\hat{\mathbf{x}}\| = 1$. It holds that $\cos \hat{\theta} \leq 1$. Thus, it follows that $\sqrt{\sum_{i=1}^{m} c_{a_i}^2} = \|\hat{\mathbf{c}}\| \geq \lambda$. □

Applied to an input vector $\mathbf{x}$ and a centroid $\mathbf{c}$, it means that the cosine similarity can only be λ or higher if the sum of the squared *centroid* values of the overlapping non-zero entries equates to at least λ^2. Based on this, we can build an indexing structure for a given minimum dot product of λ.

Given a sparse centroid $\mathbf{c}_i$ that we want to add to the structure, we first sort the index-value pairs of the present entries in $\mathbf{c}_i$ in descending order of their value. For each pair, we then perform the following steps:

1. We add the index with the centroid ID i to the general index map G. This corresponds to the basic indexing structure that was outlined at the beginning of this section. That is, for a given index, one can then retrieve a list of centroids that have a non-zero value at the corresponding position.

2. If the value is greater than or equal to λ, we immediately add the index with a *minimum overlap count* of 1 and the ID to the index map P. If a

query vector has a non-zero value at that position, it could already be enough to lead to a dot product ≥ λ, hence the overlap count of 1.

3. If the value is lower than λ, a query vector cannot reach the required minimum dot product if it only shares a non-zero entry with $\mathbf{c}_i$ at that position. We iterate through the next pairs and sum up the squared values (including the current pair), until we reach our threshold of λ^2. The number of affected pairs is then our minimum overlap count, which follows from Lemma 5.1.1. We do not need to take the previous (higher) values into account. We can assume that these pairs do not overlap with the query vector since the previously added entries to P would have already covered such a case. If we cannot reach the threshold, we stop the process for this centroid. Otherwise, we add the index with the determined count and the ID to the index map P and proceed to the next pair[2].

Given an input vector $\mathbf{x}_i$ as query and the minimum dot product of λ, we determine the list of centroids for which we need to compute the cosine similarity as follows:

1. For each non-zero entry of $\mathbf{x}_i$, we retrieve the list of overlapping centroids using the general index map G and increment our local count map C for each of the IDs in the list. Given a centroid, we can then use C to determine the number of overlapping non-zero entries with $\mathbf{x}_i$.

2. We iterate again through the entries of $\mathbf{x}_i$. For each entry, we retrieve the list of centroid candidates and the corresponding minimum overlap counts using P. We add those candidates to our resulting set that meet the minimum overlap count, which we can determine using C.

This indexing structure helps to accelerate the Spherical k-Means algorithm. During the assignment step, we first build the index from the current centroids. Given an input vector, we calculate the dot product with the centroid that corresponds to the assignment in the preceding iteration, which serves as a baseline. If the calculated similarity is at least as high as our threshold λ, we can query our structure to retrieve a (possibly) shorter list of centroids for which

[2] A naive implementation of this step would result in a worst-case time complexity of $\mathcal{O}(m^2)$ for adding a centroid with m non-zero entries. We can perform this step in linear time, though. After reaching the threshold, we save the end position. Before we proceed to the next pair, we first subtract the squared value of the current pair from the sum. For the next pair, we can then continue the summation from the previous end position until we reach our threshold. Thus, we add and subtract each squared value at most once.

we need to compute the dot product. We can guarantee that all other centroids would result in a dot product $< \lambda$.

The higher λ, the smaller the expected number of items in our centroid list, but also the smaller the percentage of input items that meet the required baseline dot product. To improve this trade-off, we build several such indexing structures with different values of λ. Upon retrieval, we select the structure with the highest threshold which is still below the respective baseline value. The general index map G has to be built only once since it does not depend on the threshold. It is possible to combine this strategy with the first one. In this case, we would just further filter the returned list of centroids according to the rules outlined in the previous section.

5.1.2 Evaluation

The approach was applied to one million tweets and 200,000 ArXiv paper abstracts, with k ranging between 50 and 5,000, to evaluate the impact of the strategies on the running time of the clustering.

Test Setup

One million English tweets and 200,000 paper abstracts (including the title) were sampled, excluding documents that only contain stop words to avoid zero vectors. The tweets originated from a bigger collection that was fetched using the Twitter API, and the papers were retrieved from ArXiv[3]. The setup script tokenizes the documents and converts them into a sparse TF-IDF-weighted Bag-of-Words representation, that is, for each present term in a document, it sets the value of the corresponding dimension to the term frequency multiplied with the logarithm of the inverse document frequency of the term. The script ignores very frequent words (stop words) and divides each vector by its length to obtain unit-length vectors. The vocabulary was not truncated, leading to 325,556-dimensional paper abstract and 783,304-dimensional tweet vectors, each containing on average 58 and 10 non-zero entries, respectively.

The evaluation compares two modes with the baseline algorithm outlined in Section 2.1.2 using three different values of k: 50, 500, and 5,000. The first mode only utilizes the non-changing clusters strategy (NCC), and the second one represents the full approach with both strategies enabled (NCC+INDEX). For the full approach, the set of minimum dot products was chosen to be $\{0.1, 0.25, 0.4, 0.6\}$. For all modes, the clustering loop terminates if the assignments do not change anymore or the centroids largely stay the same (maximum

[3] https://www.kaggle.com/Cornell-University/arxiv

Table 5.2 — The median running times of the two strategies (in minutes) on 1m tweets and 200k paper abstracts depending on different cluster sizes (lower is better). *NCC* refers to the non-changing cluster strategy and *INDEX* to the dot product indexing structure. *NCC+INDEX* represents the full approach.

	Tweets (1m)			**Abstracts (200k)**		
	$k = 50$	500	5000	$k = 50$	500	5000
Baseline	0.5	11.1	44.3	0.3	13.3	31.7
NCC	**0.4**	6.1	18.1	**0.3**	7.3	13.9
NCC+INDEX	0.7	**2.1**	**2.0**	0.6	**5.6**	**5.5**

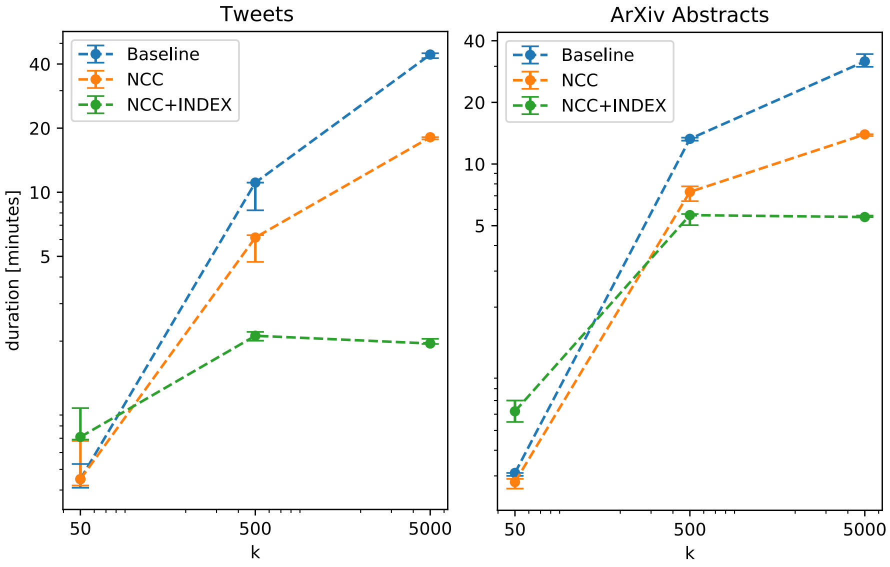


Figure 5.1 — The median running time of the two strategies, plotted on logarithmic scales. The bars indicate the interquartile range. The dots are connected with a dotted line to support the visual tracking of a specific configuration.

squared Euclidean distance between any two subsequent centroids is < 0.0001). The script ran each configuration five times on a 32-core CPU and the reported results are based on the median running time.

Results

Table 5.2 lists the results that are also plotted in Figure 5.1 on a logarithmic scale. The error bars denote the interquartile ranges. The non-changing clusters strategy leads to shorter running times across all configurations in both data sets. For larger values of *k*, the indexing structure further accelerates the clustering. The results show a more than 20-fold reduction of the time it takes to cluster one million tweets into five thousand clusters, and a more than fivefold reduction in the case of the less sparse abstracts. In contrast to the baseline scenario, clustering the documents into 5,000 instead of 500 clusters did not take more time with this approach on the two data sets. For $k = 50$, the indexing structure cannot offset the additional overhead it introduces, resulting in slightly longer running times.

Discussion

In the case of sparse input vectors, both strategies can significantly accelerate the Spherical k-Means algorithm, making a fine-grained cluster-based analysis of large document collections with $k \gg 10$ much more feasible. With a fixed number of input documents, the efficiency of the indexing structure typically increases with higher cluster sizes because the probability that a frequent word is an important component of many centroids decreases. This explains why the running time does not seem to increase above $k = 500$.

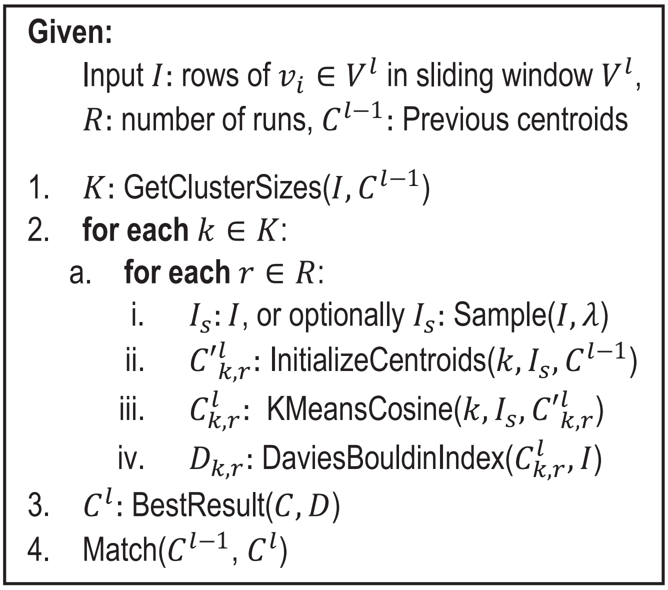
Building the index on every iteration takes time, which does not pay off for smaller cluster sizes. It is therefore advisable to enable the *INDEX* strategy dynamically whenever the number of clusters that have changed compared to the previous iteration exceeds a certain threshold (e.g., 100).

It may seem odd that the duration it takes to process 500 clusters in the baseline scenario is more than two times longer than one would expect from the running time of processing 50 clusters. One reason for this behavior is the available cache size. If the number of clusters is sufficiently small, the centroids may fit completely into the cache of the processor, reducing expensive memory fetches. For larger values of k and, thus, increased memory usage, the percentage of cache misses increases, which reduces the computational efficiency significantly.

5.2 Dynamic Spherical k-Means

Dynamic entails two important properties for the cluster-based visual analysis of streaming data. First, the algorithm should support incremental updates. The amount of visual changes on each update should be reduced and the retrieved clusters should be coherent over time. Second, the algorithm should choose a suitable number of clusters within the provided constraints (e.g., the maximum number of clusters). In addition, the algorithm needs to be fast enough as it will be invoked at regular intervals.

The high-level idea of the dynamic version of Spherical k-Means is as follows. We take the centroids of the previous run into account when we set the initial centroids so that we get more coherent clusters over time. We then run the



```
Given:
    Input $I$: rows of $v_i \in V^l$ in sliding window $V^l$,
    $R$: number of runs, $C^{l-1}$: Previous centroids

1. $K$: GetClusterSizes($I, C^{l-1}$)
2. for each $k \in K$:
   a. for each $r \in R$:
      i.   $I_s$: $I$, or optionally $I_s$: Sample($I, \lambda$)
      ii.  $C'^l_{k,r}$: InitializeCentroids($k, I_s, C^{l-1}$)
      iii. $C^l_{k,r}$: KMeansCosine($k, I_s, C'^l_{k,r}$)
      iv.  $D_{k,r}$: DaviesBouldinIndex($C^l_{k,r}, I$)
3. $C^l$: BestResult($C, D$)
4. Match($C^{l-1}$, $C^l$)
```

Figure 5.2 — Main steps of the Dynamic Spherical k-Means algorithm.

optimization with different values of k and choose the best result according to an internal evaluation criterion: the distance of each element to its corresponding centroid should be small, and the centroids should be sufficiently distinct from each other.

5.2.1 Method

Figure 5.2 outlines the main steps of the algorithm. The input I comprises the document vectors of all items in the sliding window. Compared to the previous run, some items may not be part of the set anymore, some will be new, and some may remain unchanged. Let $k_{\min}$ and $k_{\max}$ be the desired minimum and maximum number of clusters, respectively. We now have to determine which values of k we would like to test and run the clustering with (*GetClusterSizes*). If it is the very first run, we set $K = \{k_{\min}, k_{\min}+1, k_{\min}+3, ..., k_{\max}\}$. Otherwise, $K = \{k_p, k_p+1, k_p+3, k_p+6, ..., k_{\max}\}$ where k_p is the number of clusters from the previous run. We increment the step size after every step to achieve a sublinear scaling regarding $k_{\max}$.

The way we initialize the centroids $c_1, ..., c_k$ also depends on the previous run (*InitializeCentroids*). On the first run, we apply the k-means++ initialization strategy that was adapted for the cosine distance by Endo and Miyamoto [2015]: we pick one element randomly as the first centroid c_1, and for each remaining c_i we draw an element probabilistically based on its cosine distance to the nearest neighbor in the set of already chosen centroids $c_1, ..., c_{i-1}$. If all distances are zero, the current set of centroids already cover all data items, so we stop the loop early and decrease k accordingly. Hence, the initialization will never return a clustering with duplicate clusters, even if this means that $k < k_{\text{min}}$. On an incremental run, we first apply the old clustering to the new inputs and determine the set of p non-empty clusters. We then set $c_1, ..., c_p$ as the first p initial centroids and determine the remaining centroids $c_{p+1}, ..., c_k$ with the initialization strategy outlined above.

Given these initial centroids, we perform the optimization loop until convergence (*KMeansCosine*). This is analogous to the main body of the Spherical k-Means algorithm (i.e., without the initialization step). For each $k \in K$, we run the optimization process R-times to mitigate the impact of a bad initialization (the default value of R is 2).

We calculate the Davies-Bouldin-Index (DBI) (Davies and Bouldin [1979]) for each clustering result (different cluster size or run) and return the clustering with the lowest score (*BestResult*). The DBI is an internal criterion for measuring the quality of a clustering.

We now need to connect the clustering result from the previous run with the current one, that is, we need to determine which clusters stayed more or less the same, which ones are new, and which ones have been removed (because there were too many changes or redistributions to other clusters). We match a previous cluster c_i^{l-1} to a current cluster c_j^l if the majority of items associated with c_i^{l-1} that are still in I are now associated with c_j^l and there is no larger group of previous items from a different cluster for which this also holds. The remaining clusters in the current run are then classified as new, and the non-matched clusters from the previous run as removed.

The optimization loop usually converges fast, but the *optional* sampling strategy further increases the efficiency so that the algorithm is able to cluster millions of documents within seconds. One advantage of the k-Means algorithm and its variants is that we can apply any clustering to new, unseen data. Thus, we can perform the clustering on a smaller subset and extrapolate the results to the complete data set. Given a sample ratio λ, we pick $\lambda|I|$ rows randomly as input for the actual clustering run (*Sample*). However, we always use the complete data set when calculating the DBI.

Table 5.3 — Evaluation of the dynamic clustering approach on the *20 Newsgroups* data set that was processed as a batched stream. Each batch contains about 2k documents. The resulting normalized mutual information (NMI) refers to the final batch (higher is better, interquartile range in brackets). The clustering coherence score captures the average similarity of centroids between subsequent runs. The proposed approach (in bold) leads to better and more coherent clustering results compared to processing the bins on their own.

	NMI	**Coherence**	**Duration**
Baseline (distinct bins)	0.50 [0.41 - 0.50]	0.37 [0.37 - 0.38]	0.13s
10 clusters max.			
Dyn. sKMeans, 75% o.	0.62 [0.61 - 0.63]	**0.62** [0.60 - 0.62]	**0.11s**
Dyn. sKMeans, 50% o.	**0.63** [0.61 - 0.63]	0.61 [0.61 - 0.62]	0.14s
20 clusters max.			
Dyn. sKMeans, 75% o.	0.57 [0.57 - 0.58]	0.53 [0.50 - 0.53]	0.18s
Dyn. sKMeans, 50% o.	0.59 [0.59 - 0.59]	0.54 [0.53 - 0.54]	0.22s

5.2.2 Evaluation

The well-known *20 Newsgroups* data set[4] serves as a benchmark data set to evaluate the proposed Dynamic Spherical *k*-Means algorithm. The data set contains nearly 20,000 posts spread across 20 different newsgroups, and the corresponding newsgroup of each post serves as a class label to judge the quality of the resulting clustering. It is generally difficult to obtain ground truth labels since the grouping may also depend on individual preferences and the task at hand. Nevertheles, the advantage of this data set is that its labels are crowd-sourced; the authors have chosen the respective newsgroup in which they wanted to post their message.

Test Setup

The posts were ordered by their publishing date to simulate a streaming environment. Each document (that is, the *Subject* line and the actual body) was converted into a TF-IDF-weighted bag-of-words (BoW) vector representation, ignoring stop words. The inverse document frequency is calculated based on the complete data set. Each vector was normalized to have unit length. Apart from stop words, the vocabulary was not truncated. Three posts were excluded because they only contain stop words, so the final input for the clustering comprises 19,994 non-zero vectors in total.

[4] http://qwone.com/ jason/20Newsgroups/

The data set was processed in batches of approximately 2,000 posts (10% of the data set size) with two different strategies: the dynamic clustering approach and a baseline for comparison. The test script calculated the normalized mutual information (NMI) on the final batch with the corresponding labels. The NMI is an information-theoretic-based external criterion to judge how well a clustering matches the class labels. It ranges between 0 (no correlation) and 1 (perfect correlation). One advantage of this score is that it also works if the number of classes or clusters differs between the two sets. For evaluating the coherence between two clusterings and their corresponding centroid sets C_1 and C_2, the script calculated the average cosine similarity of the centroids in C_1 with their corresponding closest match in C_2 and vice versa. Each strategy and configuration was run five times to reduce the impact of outliers. The setup of each strategy was as follows:

Baseline (distinct bins): The data set was split into 10 distinct batches and the Spherical k-Means algorithm was run on each batch separately, with k set to the number of ground-truth classes in this batch. The reported *Coherence* score is the average coherence between all pairs of subsequent batches.

Proposed approach (Dyn. sKMeans): The proposed Dynamic Spherical k-Means algorithm was applied to the data set with a sliding window size equal to the batch size of the baseline scenario. As the window slides forward, new posts are added and old ones removed. The script tested two strides, one that leads to an overlap of 75% between subsequent windows and one that leads to 50% overlap. In contrast to the baseline scenario, the number of ground-truth classes was not fed to the algorithm. Rather, two configurations were evaluated with a maximum of 10 and 20 possible clusters, respectively. Here, the *Coherence* score is the average coherence between all pairs of subsequent *distinct* batches to allow a fair comparison with the baseline scores. For instance, with a stride of one-fourth of the window size (75% overlap), the script calculates the coherence between batches 1 and 5, 5 and 9, and so on.

Results

Table 5.3 lists the results. The proposed approach leads to better and more coherent clustering results in all configurations compared to the baseline, and the overall duration of each step does not increase much, despite additional optimization runs for the dynamic version. Hence, taking the previously calculated centroids into account in the initialization step of the algorithm has several benefits. It leads to more coherent clustering results between subsequent updates, it leads to faster convergence within a single optimization run, and it also leads to better clustering results on the *20 Newsgroups* data set. The higher

NMI scores may seem surprising at first, but one reason for this finding is that the initialization strategy accumulates to some extent knowledge of previous batches, which improves the generalizability on new data.

Chapter 6

Real-Time Analysis of Streaming Social Media Data

With the growing influence of social media platforms such as Twitter on society, the number of content creators, as well as the amount and topical diversity of published content on these platforms, has vastly increased. People post about their daily experiences and opinions, businesses about their new products, and researchers about their latest findings. The introduction of document visual analytics approaches in Section 2.3 shows that, apart from everyday content, social media platforms are also a valuable source for breaking developments and news, disaster management, and trading strategies. Thus, several visual analytics approaches have been developed to facilitate the needs of various domain experts, including journalists, traders, and first responders.

The sheer volume and speed of published posts pose a significant challenge that has yet to be fully addressed. Many previously published approaches only support offline analyses, offer limited analytical capabilities, or cannot handle high-volume streams. Only a few exist that support an online visual analysis of high-volume streaming data from social media. However, they either rely on additional meta-data (e.g., voluntarily shared geolocation) or extensive preprocessing (e.g., event detection). This chapter proposes a novel approach that aims to enable a visual analysis of social media streams in real-time that scales to millions of posts, without constraints on additional meta-data or extensive preprocessing.

The approach builds upon the efficient and explainable dynamic clustering algorithm that was introduced in Section 5.2 and the keyphrase extraction method ELSKE proposed in Section 3.1. The clustering algorithm groups incoming posts continuously while minimizing the amount of changes that each update would incur. The system runs two clustering processes with different levels of granularity in parallel. At the coarse level, the user interface visualizes the thematic landscape of the received posts with metaphors that are easy to comprehend and highlight what has changed after each update. Analysts can select one or more topics to retrieve more information. For such selected topics, the system continuously extracts and visualizes frequent important phrases and their relationship to each other. In addition, the fine-grained clustering process provides a digestible but diverse stream of recent posts related to this selection. Analysts can dive deeper into topics either by specifying a search query or selecting relevant clusters to start a new session that filters the stream accordingly.

The main goal is to enable adaptive visual analyses irrespectively of the volume and velocity of the stream. If analysts reside on the higher levels, they get a broad but still manageable overview of the data, and they can gradually increase the resolution to reveal more details, while still preserving their mental map.

6.1 Background

Section 2.3.3 discusses existing approaches for the visual analysis of social media posts. In the past, two main strategies have emerged to scale the analysis of streaming data.

The first strategy leverages the geolocation of published posts to structure and filter the stream. One advantage of this strategy is that posts can be processed separately for each geographic region, for instance. However, the percentage of geolocated posts has steadily declined in recent years.

The second strategy relies on an event detection algorithm to focus on specific sets of posts. On the one hand, this strategy has the advantage that the handling of dynamic changes is simplified as the analysis can then be carried out on rather static batches. On the other hand, there can also be considerable delays, which limits the usefulness of the approach for monitoring developments in real time.

StreamExplorer (Wu et al. [2018]) was one of the first published systems that made the visual analysis of non-geolocated social streams with tens of thousands of posts feasible on a budget PC. In contrast to the system that is proposed in this chapter, StreamExplorer first tries to recognize time periods of interest (events), and tweets belonging to such an event can then be clustered based on GPU-assisted self-organizing maps (SOMs). The weight vectors of the maps are initialized with the corresponding result from the previous run to create stable maps across updates. Analysts can apply several interactive lenses, for instance, the word cloud lens, to investigate areas of the map and refine the SOMs interactively. For building the tweet vector, each word is mapped to an index with a hash function to avoid a global dictionary, and the resulting vector is then projected to a lower-dimensional embedding with Random Sampling for efficiency.

The pipeline that is introduced in Section 6.3 exploits the sparsity of high-dimensional Bag-of-Words vectors and thus avoids information loss caused by reducing the dimensionality of the representations. In addition, the visualization of frequent phrases offers aggregations that are richer in context, the stream of representative posts ensures a comprehensive selection of relevant tweets, and analysts can increase the resolution of certain topics.

6.2 Task and Design Requirements

For many analysts and journalists, it is important to know what is currently happening on social media, what themes people currently talk about. This need to stay informed about major new developments is also referred to as *situational awareness*. Apart from this more explorative task, the interest in *monitoring* specific themes often increases if a major story is breaking. In such situations, it can become challenging to quickly gain an overview of what has been posted and to extract new information, despite focusing on a single theme.

Hence, the proposed approach aims to tackle two main goals: it should support both the *situational awareness* on social media and the specific just-in-time *monitoring* of currently developing themes. More specifically, the system should enable the following analytical tasks:

(T1) Overview: Analysts should gain a continuous overview of major themes people currently talk about on social media.

(T2) Details: If analysts have found an interesting theme, they should be able to learn more about it.

(T3) Monitoring: It should be possible to monitor specific themes constantly such that analysts can keep track of new developments.

(T4) Dive-in: The system should include possibilities to put certain topics at the center of the analysis and to increase the resolution of the analysis.

This approach approximates *themes* with automatically derived *clusters* based on the textual content of each post, which has three important benefits. First, the resulting clusters from topic modeling or clustering algorithms structure the content reasonably well to provide an overview and help with navigating the thematic landscape, even if they may not perfectly match the themes the analyst had in mind. Second, structuring the data with content-based clustering imposes little restrictions with regard to the data that we can process (e.g., posts do not need to be geolocated). Third, we avoid introducing additional uncertainties or delays caused by additional preprocessing such as event detection.

An important aspect of the approach is that it should support the *real-time* analysis of streaming data. As a result, it needs to deal with additional challenges compared to the analysis of static data sets. The following requirements summarize these challenges:

(R1) Efficiency: The system must rely on efficient methods that support interactive analyses on streaming data.

(R2) Flexibility: The applied methods need to quickly adapt to incoming data because we can make only little a priori assumptions about the data that we are going to process. For instance, new important terms (e.g., hashtags) may appear that we would need to consider.

(R3) Consistency: The internal state should not change too much on updates to preserve the analyst's mental map and avoid confusion.

(R4) Sparsity: The extent and frequency of visual changes should be minimized to reduce the cognitive load.

(R5) Transparency: We need to communicate not only the state but also its changes, so that users can follow and comprehend what is going on.

6.3 Architecture

Based on the requirements set out in Section 6.2, a visual analytics approach was developed to provide an overview of the currently posted content (T1), and to enable a detailed analysis of specific themes in a hierarchical manner (T2, T3, T4). The system was programmed in C# and runs under .NET 5.

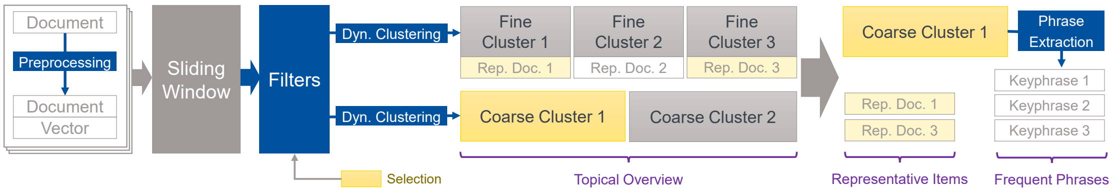


Figure 6.1 — Architecture of the approach. The system continuously collects social media posts and stores them in a sliding window. It runs two individual dynamic clustering processes in parallel. For each fine cluster, it finds the representative item and matches it to its closest coarse cluster (topic). Analysts can select one or more coarse topics. Frequent phrases in this selection will be extracted at regular intervals, and the associated representative items provide a diverse but manageable stream of relevant items. The cluster selection can act as a filter for a new session.

6.3.1 Pipeline

Figure 6.1 depicts the architecture of the approach. The system continuously receives published posts and stores them and their derived vector embeddings in a sliding window with configurable size. Each post is composed of a textual body, an optional language flag, and its publishing date. Section 6.3.2 details the preprocessing steps. The system applies Dynamic Spherical *k*-Means (Section 5.2) to all items in this window at regular intervals of about one minute.

The system establishes two parallel and independent clustering processes with different levels of granularity (i.e., different thresholds for the maximum number of clusters). By default, the first, *coarse-grained* clustering does not extract more than 10 clusters to provide analysts with an interactive *topical overview* (T1). The second more *fine-grained* process extracts up to 100 clusters per default and facilitates a diverse stream of representative posts. The upper limit for the number of main clusters was set to ten so that the interface does not exceed the usual capacity of the analyst's short term memory, but both thresholds are adjustable.

Throughout this chapter, the coarse-grained clusters are also called *topics* and the more fine-grained ones *subtopics*. It should be noted, however, that topics and subtopics do not form a classical hierarchy since both clustering processes are independent from each other. For each subtopic, the system finds its *representative item*, that is, the post closest to the respective centroid. Each post, therefore, has two cluster associations, one fine- and one coarse-grained, so each extracted representative item is also associated with exactly one topic.

Analysts can select one or more topics to retrieve additional *details* (T2). This includes a stream of representative posts that are associated with the selection, and extracted relevant keyphrases with ELSKE which was introduced in Section 3.1. Such a selection of topics can be added as a new filter, which will create a new session layer that operates on the filtered stream. Hence, with the layered approach analysts can interactively increase the resolution and adapt the specificity of their analysis (T4).

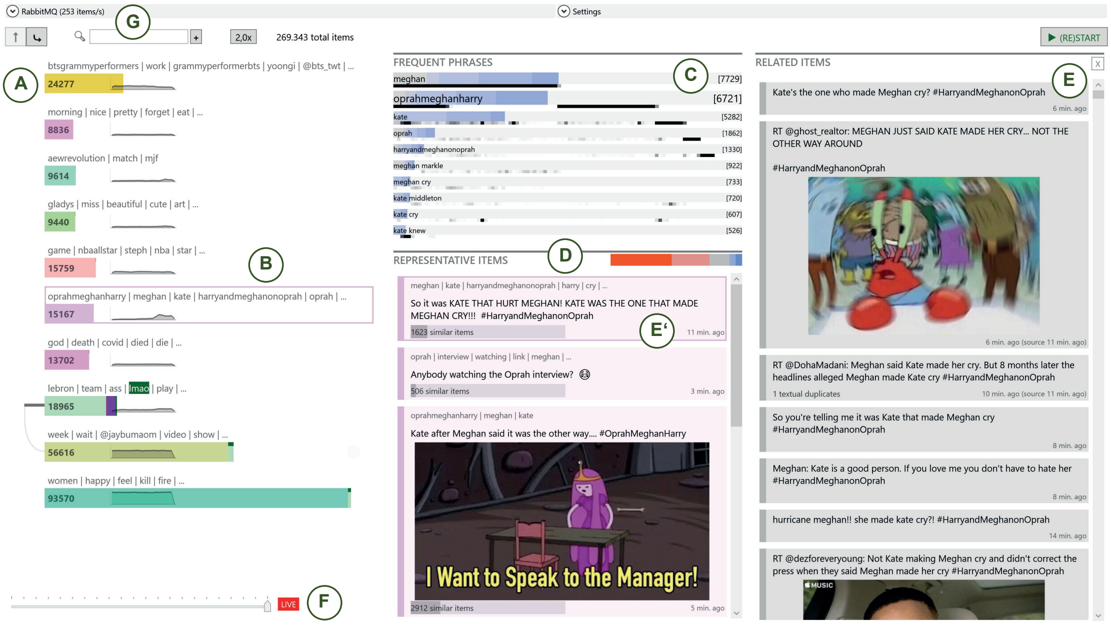
On every update, the frequent phrases will be updated and new representative posts may be added. If a new subtopic appears or the representative item of a subtopic changes and is sufficiently different, the post will be added to the stream of representative items (T3). The number of new items per update is limited because it correlates with the total number of subtopics. These items offer a *diverse* view of what is currently being posted since they originated from different clusters.

Compared to hierarchical clustering, the parallel clustering strategy ensures that both clusterings have reached their (local) minimum during the optimization; uncertainties do not accumulate across layers. Furthermore, it is more straightforward to visualize and comprehend the dynamic changes of two individual, flat clusterings compared to a more complex dynamic hierarchy.

6.3.2 Preprocessing

For each incoming tweet in the desired language, the preprocessing component creates a sparse Bag-of-Words vector representation (BoW) as input for the clustering and for determining similar tweets. It first removes URLs in the text, strips the # from hashtags, and removes the initial retweet markup if present (*'RT @Username:'*). Username mentions are preserved because they often constitute helpful context. Then, the system tokenizes the cleaned content (in lowercase) and assigns each token its corresponding vocabulary index, ignoring stop words and punctuation characters. We may need to add novel tokens to the vocabulary during this step. For the final sparse vector, we set the value of the present token indices to their corresponding TF-IDF weight, and divide the vector by its length to retrieve unit vectors. We dismiss tweets that only contain stop words to avoid zero vectors. For calculating the inverse document frequency, we use a random sample of tweets collected over several months.

One major advantage of the clustering algorithm on a set of BoW vectors is that the resulting cluster centroids can be interpreted as a weighted term list. Hence, we can easily extract the terms with the highest weight to visualize the characteristics of each cluster. The BoW approach for representing documents


RabbitMQ (253 items/s)
Settings
G
+
2,0x
269.343 total items
▶ (RE)START
A
btsgrammyperformers | work | grammyperformerbts | yoongi | @bts_twt | ...
24277
morning | nice | pretty | forget | eat | ...
8836
aewrevolution | match | mjf
9614
gladys | miss | beautiful | cute | art | ...
9440
game | nbaallstar | steph | nba | star | ...
15759
B
oprahmeghanharry | meghan | kate | harryandmeghanonoprah | oprah | ...
15167
god | death | covid | died | die | ...
13702
lebron | team | ass | lmao | play | ...
18965
week | wait | @jaybumaom | video | show | ...
56616
women | happy | feel | kill | fire | ...
93570
LIVE
F
FREQUENT PHRASES
C
meghan [7729]
oprahmeghanharry [6721]
kate [5282]
oprah [1862]
harryandmeghanonoprah [1330]
meghan markle [922]
meghan cry [733]
kate middleton [720]
kate cry [607]
kate knew [526]
REPRESENTATIVE ITEMS
D
meghan | kate | harryandmeghanonoprah | harry | cry | ...
E‘
1623 similar items
11 min. ago
oprah | interview | watching | link | meghan | ...
506 similar items
3 min. ago
oprahmeghanharry | meghan | kate
I Want to Speak to the Manager!
2912 similar items
5 min. ago
RELATED ITEMS
X
E
6 min. ago
6 min. ago (source 11 min. ago)
1 textual duplicates
10 min. ago (source 11 min. ago)
8 min. ago
8 min. ago
14 min. ago
MUSIC

So it was KATE THAT HURT MEGHAN! KATE WAS THE ONE THAT MADE MEGHAN CRY!!! #HarryandMeghanonOprah

Anybody watching the Oprah interview? 😨

Kate after Meghan said it was the other way.... #OprahMeghanHarry

Kate's the one who made Meghan cry? #HarryandMeghanonOprah

RT @ghost_realtor: MEGHAN JUST SAID KATE MADE HER CRY... NOT THE OTHER WAY AROUND

#HarryandMeghanonOprah

RT @DohaMadani: Meghan said Kate made her cry. But 8 months later the headlines alleged Meghan made Kate cry #HarryandMeghanonOprah

So you're telling me it was Kate that made Meghan cry #HarryandMeghanonOprah

Meghan: Kate is a good person. If you love me you don't have to hate her #HarryandMeghanonOprah

hurricane meghan!! she made kate cry?! #HarryandMeghanonOprah

RT @dezforeveryoung: Not Kate making Meghan cry and didn't correct the press when they said Meghan made her cry #HarryandMeghanonOprah

Figure 6.2 — Overview of the proposed system applied to a real-time stream of tweets. **A:** Topical overview of the 270k posts that are in the sliding window at the moment. **B:** Topics of interest (ToI) selected by the user. **C:** Visualization of frequent phrases in the ToI. **D:** Stream of representative posts in the ToI. **E:** List of similar posts to the selected representative post E′. **F:** History slider to peek at previous clustering versions. **G:** Dive into topics based on the search query *or* the selected topics.

is also very *efficient*. For comparison, the preprocessing pipeline can process more than 30,000 posts per second on a single core.

6.4 Visualization Techniques

As outlined in Section 6.3, the system continuously receives hundreds of social media posts each second and processes them with the parallel dynamic clustering strategy. The focus in this chapter is on tweets, but conceptually the approach would also work with textual posts from other social media platforms.

Figure 6.2 shows the user interface of the system. On the left side, the *Topical Overview* (A) visualizes the extracted topics, which will be described in Section 6.4.1. Analysts can select one or several topics of interest for additional details (B). This will activate the *Frequent Phrases View* (C) that contains a visual summary of the most important keyphrases in the selection, and the *Representa-*

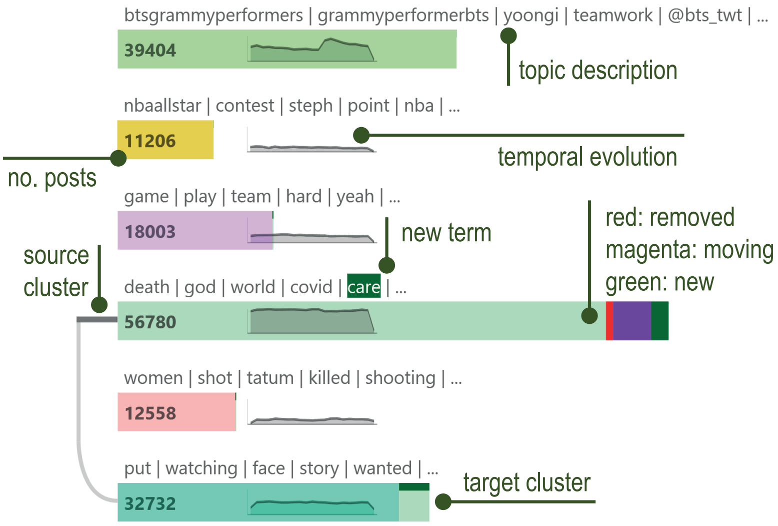

btsgrammyperformers | grammyperformerbts | yoongi | teamwork | @bts_twt | ...
39404
topic description
nbaallstar | contest | steph | point | nba | ...
11206
temporal evolution
no. posts
game | play | team | hard | yeah | ...
18003
new term
red: removed
magenta: moving
green: new
source
cluster
death | god | world | covid | care | ...
56780
women | shot | tatum | killed | shooting | ...
12558
put | watching | face | story | wanted | ...
target cluster
32732

Figure 6.3 — Overview of the topics after a new batch has been processed. The most defining terms of a topic serve as a description and the size of the cluster is mapped to the width of the bar below the description. A small line chart is embedded which shows the temporal evolution of the respective topic. Here, the cluster marked with the dark gray bar to the very left is currently being updated. The proportions of the stacked bars in red, magenta, and dark green represent the posts in the cluster that were removed, moved elsewhere, and newly added, respectively. Curved lines at the left side indicate to which clusters posts were moved.

tive Items View (D) with a stream of diverse and relevant tweets. Sections 6.4.2 and 6.4.3 discuss both views, respectively.

6.4.1 Topical Overview

The resulting topics from the coarse-grained dynamic clustering process provide analysts with an interactive overview of the various themes people currently post about. Similar to the concept of *small multiples*, the user interface plots compact summaries of the topics in a list view for an easy comparison. Figure 6.3 shows an example.

The size of each cluster is mapped to the width of its bar. We overlay the number of posts and a small line chart onto the bar. The line chart visualizes the temporal evolution of the cluster in the current sliding window, that is, the number of published posts in the cluster over time.

One advantage of the BoW model is that the cluster centroids are interpretable. If we sort the key-value pairs of a centroid vector in descending order of the value, we retrieve a list of the most defining terms of the respective cluster. We take up to five of these terms to generate a descriptive but short summary of the topic's main content, which is shown right above each bar.

Each topic has its own distinct color so that we can visually indicate changes to the clustering and easily map representative posts to their corresponding topic. We ensure that all cluster colors have roughly the same perceived brightness, for several reasons. This strategy mitigates perceived differences of the clusters due to dominant hues. In addition, it makes sure that the overlays in dark gray are clearly visible. Finally, we have a set of special colors that the system uses across all clusters to indicate the type of change after an update (e.g., dark green for new posts). These colors are darker to set them apart from the cluster colors.

On each update, the system determines what has changed compared to the previous clustering, for instance, which posts have moved from one to another cluster. However, revealing all changes at once might lead to a sensory overload. Thus, the changes are visualized cluster-by-cluster from top to bottom. A short thick line in dark gray to the left of the bar marks the current *source cluster* of the update. For instance, in Figure 6.3, the topic with *'death'* as the most defining term is currently being updated.

If a new term appears in the topic description, it is highlighted in dark green for some seconds. We further replace the bar of the current source clusters with a stacked bar to indicate the proportion of posts in the cluster that were removed from the sliding window in red, posts that have been moved to other *target topics* in magenta, new posts in dark green, and the remaining posts in the original color of the cluster. For each target topic, we append a bar that represents the proportion of posts which have been moved from the source to the respective cluster. This bar has the same color as the source cluster, but with a dark green line at the top.

We also visualize the flow to the prevailing target topics with curves on the left side of the list so that analysts can quickly spot if clusters split or merge. Both the thickness *and* the gray level of a curve are proportional to the square root of the number of moved posts the curve should represent. In theory, there can be as many curves as there are topics (minus one), so we have to limit the maximum thickness. As a result, depending on the visual encoding we would either have very thin or very light curves at times, so we use both visual variables to encode a wider range of values.

The duration of each visual update varies depending on the complexity. The more affected target clusters and the more appearing terms, the longer we wait before we proceed to the next cluster. The more visual changes, the longer users might need to grasp them. Analysts can adjust the average speed to their needs with a toggle button at the top of the window, similar to changing the playback speed of a video. The changes are rolled out step by step but without animated transitions. This strategy leverages visual preattentive processing so that users can immediately notice outliers and compare changes across steps more accurately.

After each update, we save the state of the topical overview in the history. Users can choose with a slider at the bottom left of the window whether they want to peek at a previous version of the topical overview (Figure 6.2 F). For instance, this is handy when they cannot monitor changes continuously.

Analysts can enter a search query above the list of topics (Figure 6.2 G). Then, a new clustering session starts in which only the posts that match the query are processed. Similarly, analysts can select one or several topics as a filter. Both types of filters can be chained to increase the resolution down to a handful of posts. However, only the clustering processes from the current layer are actively running. For instance, if an analyst dove into a topic, we create a filter based on the current set of centroids at the parent layer and we use that filter for the new session, but the clustering processes of the parent level will then pause and only continue their work if the analyst goes back to the parent session.

6.4.2 Frequent Phrases View

The short lists of terms already hint at what each topic is about, but they only offer little context. When analysts select one or several topics of interest, the goal of the approach is to visualize more precisely which issues and themes to what extent people are tweeting about in this topic selection. To achieve this, we continuously extract the most relevant keyphrases from all tweets belonging to the selection and visualize their distribution across the posts as well as their temporal evolution.

Figure 6.4 depicts an example of the top ten phrases in a topic related to the super bowl event, sorted by the frequency in descending order. The number of tweets containing the respective phrase is shown at the very right of each row. The font size of the phrase correlates with the returned importance score from the keyphrase extraction algorithm. If the phrase has just appeared after an update, it will be highlighted in dark green for some seconds to catch the attention of the analyst.

The stacked bar, composed of five bins from dark gray to blue, represents the proportion of tweets containing the phrase compared to all tweets in the selection. Each bin corresponds to one fifth of the sliding window time range and depicts the proportion of tweets that were originally published within

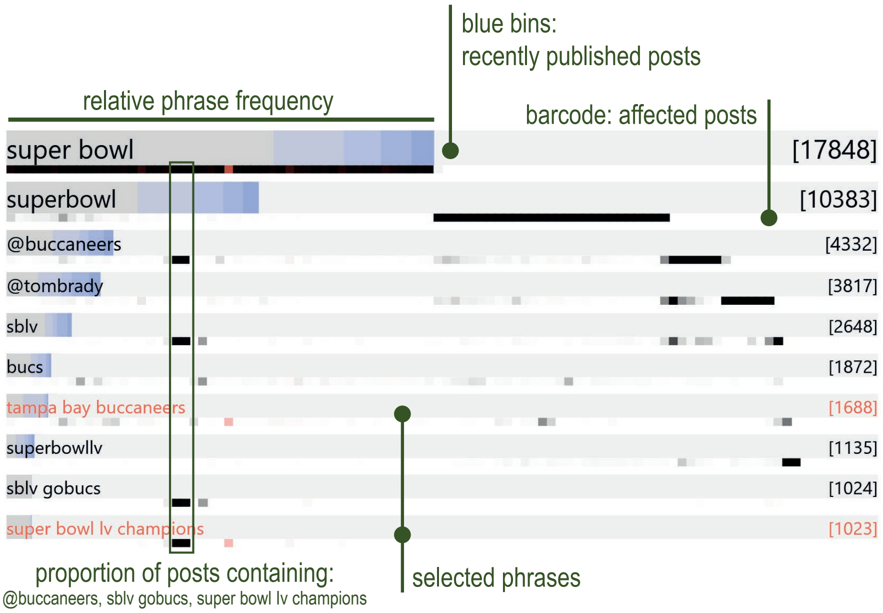


Figure 6.4 — Visualization of the most important frequent phrases in a selection of topics. The bar composed of dark gray and blue-ish bins represents the number of posts containing the respective phrase. The blue bins depict the proportion of *new* posts to indicate trends. Each post is mapped to a horizontal position in the barcode-like visualizations below each phrase. Dark ticks indicate the posts containing the respective phrase so that users can see which set of phrases often co-occur. Analysts can select phrases to highlight the overlap in orange.

that time frame (i.e., the effective date of any retweet is the publishing date of the original tweet). For instance, in a sliding window of 20 minutes, the blue bin at the right represents how many tweets have just been published within the last four minutes, and the dark gray bin to the left how many are older than 16 minutes. In the example, the bars of the two phrases at the bottom are largely gray whereas the bar corresponding to *'@tombrady'* is largely blue. This means that people are actively tweeting *new* posts containing *'@tombrady'* at the moment but seldomly ones with *'super bowl lv champions'* (except for retweets).

Right underneath each phrase, there is a small barcode-like strip composed of 100 bins that visualizes the distribution of the corresponding phrase. This strip should enable analysts to quickly estimate which phrases appear in the same tweets. Let n be the total number of posts from which the phrases were extracted. Then, we assign each post a unique integer in the range $[1, n]$ and rescale these to real numbers in the range $[0, 100)$. The first bin then corresponds to all posts with a value in $[0,1)$, the second bin to all in $[1,2)$, and so on. The shade of each bin from white to black represents the proportion of posts in that bin containing the respective phrase. For instance, most of the tweets in Figure 6.4 contain *'super bowl'* or *'superbowl'*, but rarely both because there are only a few darker areas at the same horizontal position. In other words, the intersection of the first two strips would be mostly white.

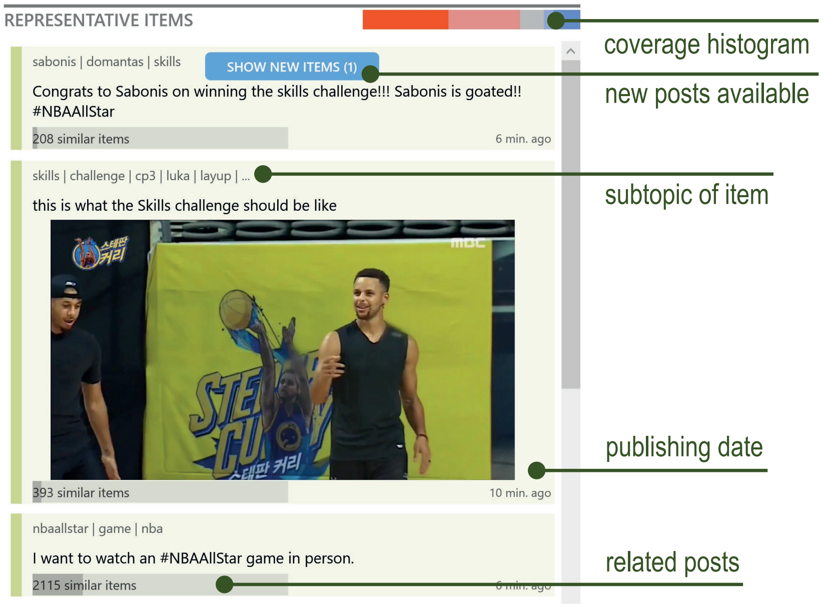
We greedily optimize the mapping to increase the length and number of contiguous blocks in black. Starting with the most frequent phrase, we assign consecutive numbers to all unassigned tweets containing a certain phrase. Let us assume we have three phrases p_1, p_2, p_3 (in descending order of the frequency) and 100 tweets in total. 40 tweets (A) would contain the first phrase, 20 (B) p_2 but not p_1, and 10 (C) p_3 but neither p_2 nor p_1. Then, we would assign the 40 tweets (A) unique numbers from 0 to 39, tweets in B from 40 to 59, tweets in C from 60 to 69, and the remaining 30 tweets would be assigned to slots 70 to 99. Hence, tweets containing p_2 would never be assigned a number higher than 59, but they could get a number lower than 40 if they also contain p_1.

Analysts can click on phrases to select them. The tweets containing all selected phrases will be highlighted in orange, and this filter then also applies to the stream of representative posts that is described next.

6.4.3 Stream of Representative Posts

The main purpose of the fine-grained clustering process is to extract representative posts for each subtopic and map them to their corresponding topic, as discussed in Section 6.3. Inspired by the concept of *learning by example*, these posts should convey the variety of points that are currently being discussed in a topic of interest. As individual posts, they are richer in context, but they should also cover different aspects because they originated from different subtopics. It should be noted that the number of new representative posts per update is bound by the total number of subtopics by design. Hence, semantically zooming into topics does not increase or decrease the average number of posts in the stream, it only leads to a better coverage of the more specified topics.

Figure 6.5 shows an exemplary stream of posts related to a selected topic. The term list at the top of the post describes the associated subtopic. For instance, the first tweet in the example that congratulates the basketball player for winning the skills challenge belongs to the subtopic that has *'sabonis'*, *'domantas'*, and *'skills'* as the most defining terms. The color of each tweet relates to the topic it belongs to, in case analysts select more than one topic. At the bottom, a small bar chart depicts the proportion of tweets in the selection that are similar to the representative post. The similarity is calculated based on the cosine similarity



sabonis | domantas | skills

Congrats to Sabonis on winning the skills challenge!!! Sabonis is goated!! #NBAAllStar

208 similar items 6 min. ago

skills | challenge | cp3 | luka | layup | ...

this is what the Skills challenge should be like

393 similar items 10 min. ago

nbaallstar | game | nba

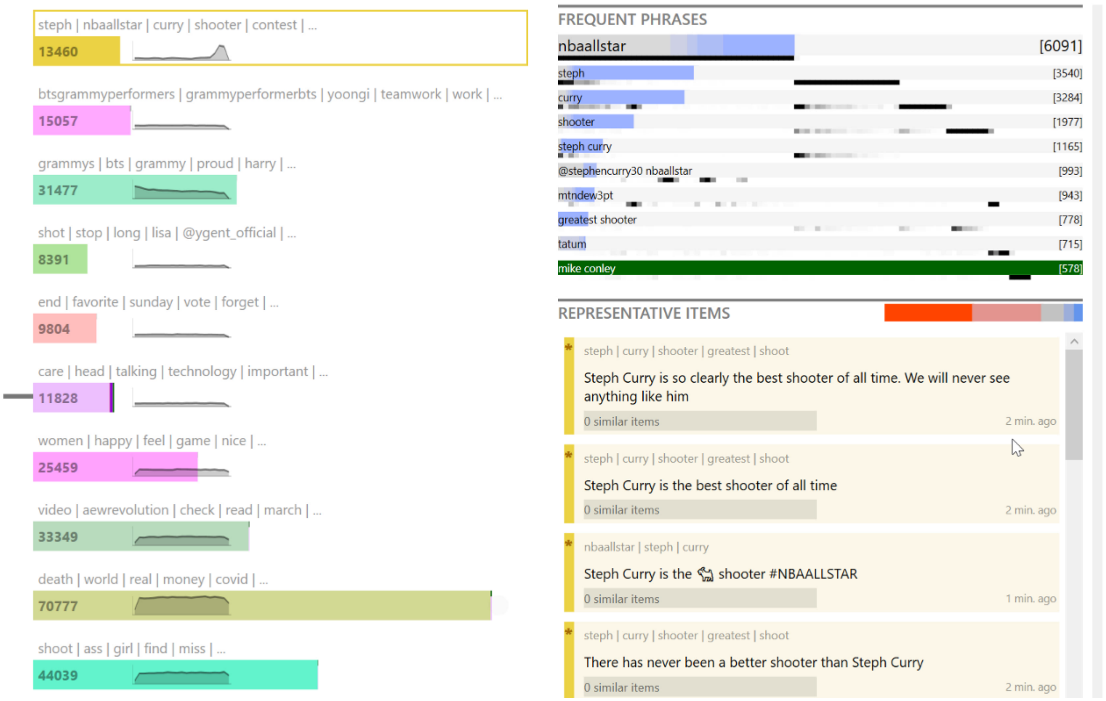
I want to watch an #NBAAllStar game in person.

2115 similar items 6 min. ago

Figure 6.5 — Stream of representative posts related to a selection of topics. The color of the associated topic is mapped to the background of each tweet. The list of terms at the top of each item describes the subtopic of that item. Analysts can click on the bar chart that shows the proportion of similar posts to retrieve a list of such similar posts in a new column. The coverage histogram at the top-right corner composed of five stacked bars indicates how well the items in the list cover all posts within the topic selection (red: proportion of similar tweets, blue: dissimilar tweets).

between the document vectors. A click on the bar will open a separate column with a list of related tweets, sorted by their similarity to the reference item.

If new posts have been extracted after an update, a blue button to insert these posts appears as a notification to the user. At the top-right of the view, next to the header, stacked bars from red to blue visualize how well the extracted representations cover all posts in the selected topics (coverage histogram). For each post in the topic selection, the system calculates the cosine similarity to the representative post of the subtopic they are in. The width of a bar then corresponds to the number of posts that have a cosine similarity in a certain range. Red represents posts with a high similarity, whereas the blue bar the ones with a low similarity, which are thus hardly covered by the stream of representative items. If the stacked bars are mainly blue-ish, this indicates that the topic selection relates to a very diverse set of themes which are not adequately represented by the subtopics. Analysts should then consider to increase the resolution of the analysis with a new filtered session layer. The



steph | nbaallstar | curry | shooter | contest | ...	13460
btsgrammyperformers | grammyperformerbts | yoongi | teamwork | work | ...	15057
grammys | bts | grammy | proud | harry | ...	31477
shot | stop | long | lisa | @ygent_official | ...	8391
end | favorite | sunday | vote | forget | ...	9804
care | head | talking | technology | important | ...	11828
women | happy | feel | game | nice | ...	25459
video | aewrevolution | check | read | march | ...	33349
death | world | real | money | covid | ...	70777
shoot | ass | girl | find | miss | ...	44039

FREQUENT PHRASES

nbaallstar	[6091]
steph	[3540]
curry	[3284]
shooter	[1977]
steph curry	[1165]
@stephencurry30 nbaallstar	[993]
mtndew3pt	[943]
greatest shooter	[778]
tatum	[715]
mike conley	[578]

REPRESENTATIVE ITEMS

steph | curry | shooter | greatest | shoot
Steph Curry is so clearly the best shooter of all time. We will never see anything like him
0 similar items 2 min. ago

steph | curry | shooter | greatest | shoot
Steph Curry is the best shooter of all time
0 similar items 2 min. ago

nbaallstar | steph | curry
Steph Curry is the 🐐 shooter #NBAALLSTAR
0 similar items 1 min. ago

steph | curry | shooter | greatest | shoot
There has never been a better shooter than Steph Curry
0 similar items 2 min. ago

Figure 6.6 — The overview of the topics on the left side shows that people are currently tweeting about the Grammys, the NBA All Stars Event, Covid, and the AEW Revolution Wrestling Event, among other more diverse topics. The notable uptick in the small line chart of the selected topic indicates that this cluster of posts about the basketball player Steph Curry has just gone viral. On the right side, the background gradients of the frequent phrases in this topic are mostly blue-ish, which also indicates that many users publish new tweets containing these phrases.

stacked bars in Figure 6.5 are largely red, though, so here, the posts seem to cover most of the current content in this topic.

6.5 Use Cases

6.5.1 NBA, BTS, and Oprah

This use case is about tweets that were streamed on Sunday evening (US ET), March 7th, 2021, at a rate of about 250 new posts per second. The size of the sliding window is 20 minutes, corresponding to roughly 250,000-300,000 tweets.

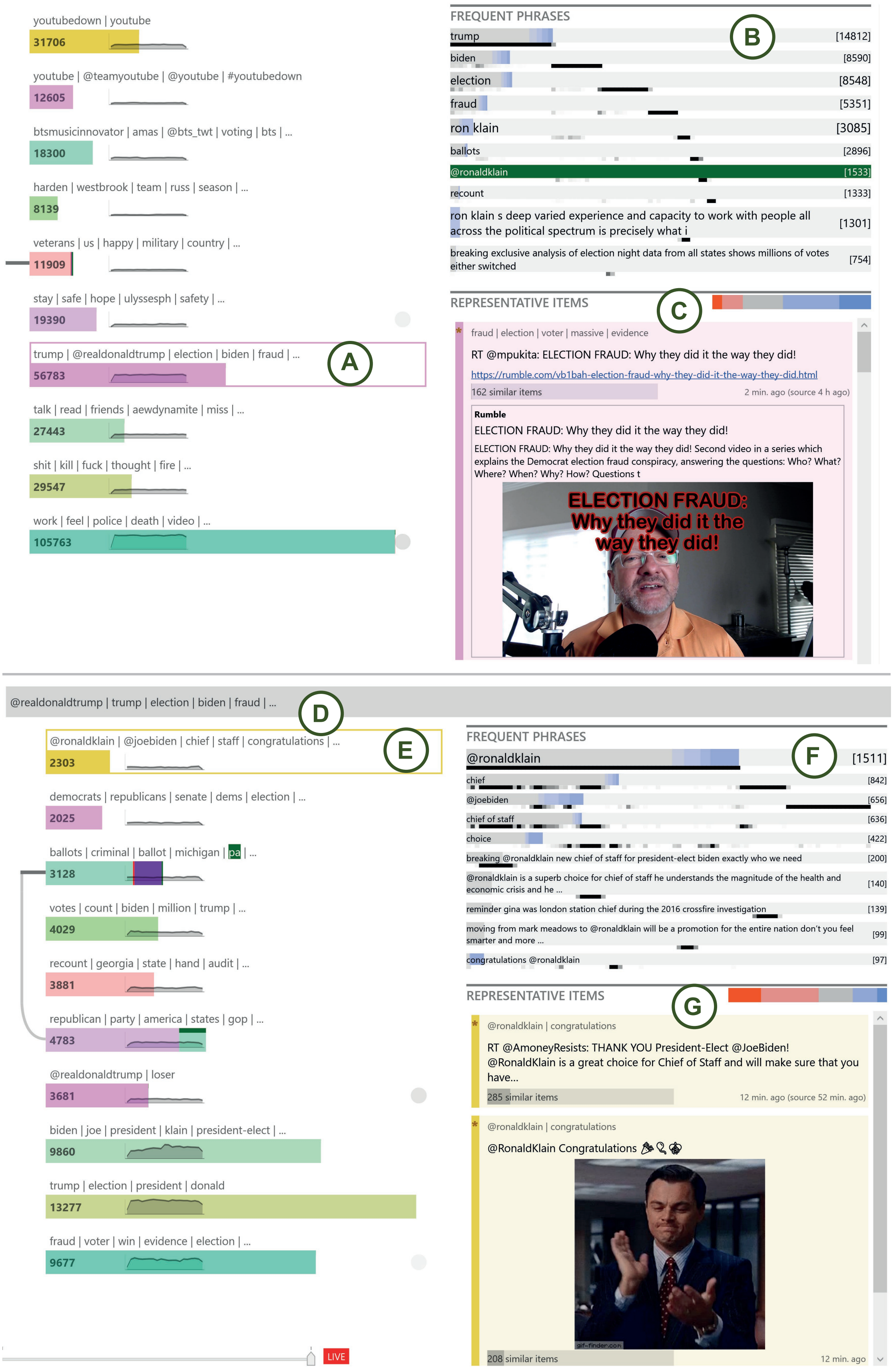
The analyst first scans all topics and notices that people currently seem to tweet about the Grammys, the NBA All Stars Event, Covid, the AEW Revolution

Wrestling Event, and other more general topics. One bigger topic that includes the hashtag *'btsgrammyperformers'* catches the interest of the analyst who clicks on it to find out more. From the visualized distribution of the frequent phrases they conclude that this is a homogeneous cluster in which people mostly tweet about the fact that the boygroup BTS was announced to perform at the Grammys the week after, including celebrations of Min Yoon-gi who is a member of BTS (*'what yoongi wants, yoongi gets'*).

The analyst notices that the line chart of the *'steph, nbaallstar, curry'* cluster indicates a strong upward trend (Figure 6.6). After switching to that topic, the list of representative items reveals that thousands of the received tweets cheer on the basketball player Steph Curry. After a while, the blue badge appears, notifying the analyst that new posts are available. They click on it to insert the recently extracted tweets into the list. Most of the new tweets congratulate him (*'Steph Curry greatest shooter ever!'*), again with thousands of similar tweets, indicating that the basketball player is performing very well in the currently running game.

Later, a new cluster appears about Meghan Markle at Oprah (*'harryandmeghanonoprah'*). Initially, people talk about whether and where to watch the interview. Shortly afterward, the line chart of the topic goes strongly up (Figure 6.2). From the visualization of the frequent phrases (C), the analyst concludes that the social media community has just started to increasingly post about Meghan giving an interview to Oprah because the phrases have largely blue-ish backgrounds. One representative post (E') talks about an utterance in the interview that it was apparently Kate who made Meghan cry. The analyst clicks on the bar in the lower-left to retrieve related tweets (Figure 6.2, right column). From the tweets, it becomes clear that most users make fun of the fact that Meghan goes after Kate, even though some seem to be very upset (*'It's Kate I'm sure'*). Later on, some tweets have a more serious tone after Meghan talks about racism (*'How dark would Archie's skin be?!!'*).

This use case shows that the proposed system not only makes analysts aware of major topics that are discussed on social media, but it also enables the specific monitoring of ongoing events, even if the frequency of posts suddenly increases. It also shows that the clustering-based approach helps to differentiate between several major events happening at the same time. In addition, the topic descriptions clearly indicate that considering novel terms in the pipeline (e.g., *'btsgrammyperformers'*) is beneficial for the clustering of tweets.



youtubedown | youtube
31706

youtube | @teamyoutube | @youtube | #youtubedown
12605

btsmusicinnovator | amas | @bts_twt | voting | bts | ...
18300

harden | westbrook | team | russ | season | ...
8139

veterans | us | happy | military | country | ...
11909

stay | safe | hope | ulyssesph | safety | ...
19390

trump | @realdonaldtrump | election | biden | fraud | ...
56783

talk | read | friends | aewdynamite | miss | ...
27443

shit | kill | fuck | thought | fire | ...
29547

work | feel | police | death | video | ...
105763

FREQUENT PHRASES

trump	[14812]
biden	[8590]
election	[8548]
fraud	[5351]
ron klain	[3085]
ballots	[2896]
@ronaldklain	[1533]
recount	[1333]
ron klain s deep varied experience and capacity to work with people all across the political spectrum is precisely what i	[1301]
breaking exclusive analysis of election night data from all states shows millions of votes either switched	[754]

REPRESENTATIVE ITEMS

fraud | election | voter | massive | evidence

RT @mpukita: ELECTION FRAUD: Why they did it the way they did!

https://rumble.com/vb1bah-election-fraud-why-they-did-it-the-way-they-did.html

162 similar items — 2 min. ago (source 4 h ago)

Rumble

ELECTION FRAUD: Why they did it the way they did!

ELECTION FRAUD: Why they did it the way they did! Second video in a series which explains the Democrat election fraud conspiracy, answering the questions: Who? What? Where? When? Why? How? Questions t

ELECTION FRAUD: Why they did it the way they did!

@realdonaldtrump | trump | election | biden | fraud | ...

@ronaldklain | @joebiden | chief | staff | congratulations | ...
2303

democrats | republicans | senate | dems | election | ...
2025

ballots | criminal | ballot | michigan | pa | ...
3128

votes | count | biden | million | trump | ...
4029

recount | georgia | state | hand | audit | ...
3881

republican | party | america | states | gop | ...
4783

@realdonaldtrump | loser
3681

biden | joe | president | klain | president-elect | ...
9860

trump | election | president | donald
13277

fraud | voter | win | evidence | election | ...
9677

FREQUENT PHRASES

@ronaldklain	[1511]
chief	[842]
@joebiden	[656]
chief of staff	[636]
choice	[422]
breaking @ronaldklain new chief of staff for president-elect biden exactly who we need	[200]
@ronaldklain is a superb choice for chief of staff he understands the magnitude of the health and economic crisis and he ...	[140]
reminder gina was london station chief during the 2016 crossfire investigation	[139]
moving from mark meadows to @ronaldklain will be a promotion for the entire nation don't you feel smarter and more ...	[99]
congratulations @ronaldklain	[97]

REPRESENTATIVE ITEMS

@ronaldklain | congratulations

RT @AmoneyResists: THANK YOU President-Elect @JoeBiden! @RonaldKlain is a great choice for Chief of Staff and will make sure that you have...

285 similar items — 12 min. ago (source 52 min. ago)

@ronaldklain | congratulations

@RonaldKlain Congratulations 🎉🍾🎊

gif-finder.com

208 similar items — 12 min. ago

Figure 6.7 — Top: top-level view with one selected topic (A) and the corresponding frequent phrases (B) and representative posts (C). **Bottom:** sub-level view using filtered session (D) based on top-level topic (A), with selected topic (E) and its frequent phrases (F) and representative posts (G).

6.5.2 YouTube Outage and Dive Into Politics

This use case deals with streamed tweets from November 11th, 2020. The top part of Figure 6.7 depicts the top-level view. The topic description of one of the first topics that catches the interest of the analyst contains only two major terms: *'youtubedown'* and *'youtube'*. The analyst hypothesizes that YouTube is experiencing some kind of outage and selects the topic to investigate their hypothesis. The visualization of the frequent phrases confirms that nearly all of the more than 30,000 posts in this topic indeed contain *'youtubedown'*. A large proportion of the phrase background is composed of blue-ish bars, which indicates an ongoing issue since many people are actively posting new tweets and not just retweets. The analyst notices that there is another similar topic about *'youtube'* and adds it to the current selection. The combined stream still contains only a handful of posts, but the large red bar in the coverage histogram next to the header indicates that these posts cover most of the published tweets well. Thus, they conclude that most posts are slight variations of the utterance that YouTube is down.

The analyst now briefly scans some of the remaining topics. Apart from the outage, people celebrate BTS for winning an award, tweet about NBA players Harden and Westbrook because Russell Westbrook apparently wants to leave the Rockets, wish everyone a happy veteran's day, and pray for the safety of people affected by the Typhoon Ulysses that struck the Philippines.

Then, the analyst looks at the topic about Biden, Trump, and the presidential election (Figure 6.7 A). However, the coverage histogram (C) reveals that the topic is relatively diverse because the stream does not adequately represent a significant proportion of posts. They click on the Dive-In button which is situated at the upper-left of the window to start a new filtered session. The Topical Overview now relates to topics derived from clusters of the parent topic, as shown in the bottom part of Figure 6.7. The analyst first focuses on the *'@ronaldklain'* topic (E) and learns that Joe Biden has picked Ronald Klain as Chief of Staff. There are many congratulations (G), but also critical remarks about his alleged handling of the swine flu.

It should be noted that the visual patterns between the frequent phrase visualization of the top-level topic (B) and the sub-level topic (F) clearly differ. Whereas the step-like pattern in (B) reveals that the different keywords seldomly appear together, the pattern in (F) shows significant overlaps of keyphrases and, thus, points toward a more homogeneous topic. Furthermore, the proportion of the two red-like bars in the coverage histogram (G) is much larger compared to (C).

The analyst now skims through the other topics in which people talk about the ballots (*'many faked ballots in detroit'*, *'Where ARE those ballots?'*), alleged voter fraud, the recount by hand in Georgia that the secretary of state seems to have just announced, but also Democrats celebrating the win.

This use case demonstrates that the visualization of frequent phrases and the coverage histogram enable analysts to assess the composition of topics and the diversity of relevant posts. It also highlights the utility of the layered approach for semantically zooming into topics of interest.

6.6 Discussion

Based on the AIX process (Section 2.5), the proposed approach applies the visually explainable dynamic clustering algorithm (introduced in Section 5.2) to scale the real-time analysis of streaming social media posts. The clustering visually structures the data efficiently without the need of additional metadata or any kind of pre-filtering, facilitating the continuous monitoring of developments with minimal delays. The system was successfully tested with a sliding window size of more than two million posts, which confirms that it can handle a large number of posts even on a budget PC. The transparency of the clustering algorithm allows analysts to comprehend what each topic is mainly about and how the they evolve over time.

The approach offers further advantages compared to previous work. For a selection of topics, analysts can retrieve a real-time stream of representative posts irrespectively of the actual frequency of published posts. Thus, the system scales not only in terms of the number of posts it can process but also regarding the visual mappings. With the visualization of the frequent phrases, analysts can assess in greater detail what the topic is about, how homogeneous it is, and whether it is currently going viral. Furthermore, the approach is agnostic as to which social media platform is analyzed because it only processes the textual content of the posts and their publishing date. The system was also tested on tweets with different languages, including Spanish and German, to verify that it generalizes to non-English languages.

The system also has limitations. The pipeline is purely content-based and enables top-down analyses, so analysts may miss smaller developments with just a handful of associated posts and retweets. The system could be extended with an anomaly detection algorithm in the future so that users can be notified of smaller developments that exhibit an unusual posting trend. However, given the huge number of posts that are published every minute, additional constraints are probably necessary to limit the frequency of detected anomalies.

Another limitation is that the dynamic clustering algorithm currently ignores media content for efficiency reasons. People increasingly post images or videos with only a short description, which reduces the effectiveness of the clustering.

One important aspect regarding the suitability of the approach is the homogeneity of the posts. In the case of major events, a large proportion of published posts belong to these events (e.g., significant football matches or elections). The everyday content is much more diverse, though. Hence, it may be difficult to group such posts in a meaningful way on a coarse level. Analysts would then either need to increase the maximum number of topics or dive into one of the less focused topics to find interesting themes.

Users might also want to include tweets from different languages regarding one specific ongoing event. The system could be extended such that it determines the inverse document frequency from different reference corpora, based on the language of each tweet. However, posts in different languages would most likely be assigned to the same cluster only if they shared some defining words (e.g., same hashtags).

In summary, this chapter proposed an interactive system for the visual analysis of streaming social media posts. The use cases indicate that the system not only supports analysts in getting an overview of what is currently happening on social media platforms but also in monitoring specific topics at different resolutions. Compared to previous work, the approach enables a fast and comprehensive analysis of larger data sets in real-time and, thus, contributes to making the visual analysis of streaming documents more scalable.

Chapter 7

Multivariate Analysis with Visual Neural Decomposition

The previous chapters introduced new techniques for providing interactive overviews of large document collections and means to analyze more specific topics of interest for both static and dynamic data. However, analysts may also want to understand how words and phrases, recognized entities (e.g., person mentions), and associated attributes (e.g., publishing date of an article) correlate with one another. We can generalize this problem to visual multivariate analysis of multidimensional datasets. Apart from text data, many interesting real-world data sets for which we want to analyze relationships between variables have more than three dimensions, for instance, socio-demographic data to analyze consumer or voting behavior, meteorological measurements to predict the weather, or integrated circuit (IC) testing measurements to analyze the performance of chips. Unfortunately, analyzing multivariate data sets is a challenging task, especially if the number of variables grows well above three. One common goal analysts often face is to explore whether and how exactly the different variables influence a specific variable of the data set, the so-called *target variable*. This problem corresponds to performing a multivariate analysis with one dependent and many independent variables.

Section 2.4 introduced the general concept and diverse challenges of a multivariate analysis. It is particularly cumbersome to visually analyze large data sets with many attributes due to the curse of dimensionality and the limits of human perception. We need powerful methods that can recognize multidimensional relationships, but they also need to be efficient enough for enabling interactive analyses. In addition, we need to think about user interfaces and visual mappings that scale to many variables and large data sets. Existing approaches have several shortcomings. They either do not scale to large data sets with many variables, apply simpler (e.g., linear) methods for extracting relationships, require an iterative workflow, or focus on up to trivariate relationships.

This chapter proposes an efficient method for visually explaining multivariate data sets that is based on the AIX framework introduced in Section 2.5. It scales to hundreds of variables and is able to recognize nonlinear, multidimensional relationships. The goal of the approach is to extract and visualize cases that lead to high values of a specified target variable. The main idea is to *decompose* the problem into individual problem cases using an interpretable neural network architecture, so that each case can be visualized and analyzed separately. The output of the model guides analysts to thoroughly explore interesting combinations of variables that correlate with the target.

Crucially, the behavior of the target variable should be explained with the original features, the remaining (or independent) variables of the data set, because they often have a semantic meaning for analysts (Sedlmair et al. [2014]). Thus, we need a visually interpretable, scaleable technique that can recognize non-linear correlations between input variables and a specified target variable of the data set. To achieve this, we first train a neural network using a novel architecture to predict the target variable based on the input of the other variables. The neural network decomposes the approximated function of the target variable into components, the *neural* nodes in the hidden layer. Then, the behavior of these individual neurons is visualized to explain separately for each case in which conditions the target variable adopts high values. Several interaction possibilities let analysts explore the found relationships in deep. As mentioned in Section 2.5, the approach exploits machine learning techniques not for the sake of making predictions, but rather to learn something about the source, the data set on which the model was trained.

7.1 Background

Visual Neural Decomposition (VND) explains the behavior of a dependent variable (the target variable) with the remaining independent variables of the set, assuming that these variables have a semantic meaning for analysts (e.g., age). This is in some ways similar to *input-output models* in the context of visual parameter space analysis (Sedlmair et al. [2014]), but the approach does not aim to develop a systematic understanding of the complete input space.

Godfrey and Gashler [2017] introduced *Neural Decomposition* as a new neural network technique to decompose time-series data into sums of periodic and nonperiodic components. Similarly, the target variable is decomposed into sigmoidal components based on the input variables, but the proposed approach in this chapter differs regarding its goal and underlying technique. VND uses traditional neural networks with a novel regularization technique. The main idea is to *visualize* the learned decomposition to understand which conditions lead to high values of the target variable.

Several approaches have been proposed in the literature that use different strategies to scale the analysis. Section 2.4 provides an overview of related work. Some techniques initially reduce the dimensionality of the data set to ease the handling. However, this also means that subsequent processing steps only work with the reduced data which often contain less information compared to the high-dimensional input. Iterative and explorative approaches typically assume that interesting relationships are already apparent in a small subset of the affected variables, but Section 7.5.1 shows that this is not generally true. Another strategy is to apply bivariate statistics or simpler models for deriving correlations, which may miss nonlinear or multidimensional correlations.

More similar to the proposed approach are visual techniques that use decision trees to model the data. For instance, Neto and Paulovich [2021] convey the paths of a random decision tree forest in a matrix-like visualization. Smaller trees are easy to comprehend, but continuous relations are difficult to model and more complex patterns easily lead to deep trees or larger ensembles, which makes it difficult for analysts to gain a thorough understanding of the data set.

VND performs the analysis on the original, high-dimensional data set and is able to recognize more complex relationships that iterative approaches would miss. The fitting process scales to larger data sets since the training objective is optimized using batched stochastic gradient descent. Based on the model, each decomposed case can be analyzed separately in an interactive visualization that is specifically tailored to the respective case, for instance, with an individual importance ranking of the affected variables.

7.2 Requirements

Experts from a wide range of domains have to deal with the challenges of analyzing multivariate data sets. A case study with a domain expert and research fellow of one of the world's largest provider of automatic integrated circuit (IC) test equipment was conducted to learn more about the requirements of the approach when applied to real-world data in a typical engineering domain.

In post-silicon validation, produced IC chips are tested under varying conditions to detect design bugs and to reveal sensitivities of a target variable with respect to other variables as hints for design improvements (Mitra et al. [2010]). The automated testing of the chips often produces large high-dimensional data sets with up to hundreds of variables that capture parameters, environmental factors, and resulting error measurements. Engineers analyze the data sets to find out which parameter range combinations may lead to undesired behaviors of the chip (out-of-spec behavior), and to determine parameters for optimal performance (tuning), for instance, to maximize battery life of the chip.

While the total number of variables is high, the actual number of relevant variables for a specific case is often less than half a dozen, according to the expert. However, each data set typically exhibits several cases involving different variables, for instance, several bugs relating to different environmental factors. Some of these patterns may only appear in a small fraction of the data set.

The expert also stated that most of their problems at hand can be formulated such that the goal is to find conditions under which a certain variable (e.g., error rate) has high values. As of now, the analysts and engineers make heavy use of scatter plot matrices to investigate possible correlations between several subsets of variables and the selected target variable, which is time-consuming.

7.3 Method

Given a multivariate data set, analysts should be able to investigate in which cases (regarding the independent variables) a specified target variable y adopts values in a specified range. The goal of the proposed method is to visualize such cases separately to provide digestible visual explanations, while still supporting more complex relationships between y and the remaining variables of the data set.

As stated in Section 2.5, multivariable linear regression is often used to infer which independent variables x_i seem to have the most influence on the dependent variable y. While the resulting coefficients are easy to interpret, linear regression cannot capture more complex correlations involving nonlinear relationships or combinations of variables (e.g., y is only high if two input variables are both high). Neural networks, on the other hand, can approximate highly complex functions, but are also more difficult to interpret.

The motivation to use neural networks for extracting interesting relationships in the data set stems from the fact that, in simplified terms, a single-output neural network essentially combines several *localized* quasi-regressions as represented by the hidden nodes. It is therefore likely that some of these nodes capture specific cases that lead to values of y in the desired range, and we can convey this information when we access and visualize the behavior of such a node. While the behavior of each node in the context of the network is more powerful than a linear regression, it is still less complex than understanding the whole network at once.

The main idea of the approach is to first train a neural network predicting the target based on the other variables in the data set (*input variables*). Afterward, the hidden nodes' activations are used to visually explain how certain parts and variables of the data set correlate with the target. To make this work, a novel regularization technique is introduced (Section 7.3.4) that encourages the neural network to model the relationships in a way that is easier to interpret. In other words, the approach leverages the training process to perform multiple nonlinear regressions, and analysts can then further explore these in the provided interactive visual interface.

The resulting accuracy of the network is less relevant since we are mainly interested in how the model captures the underlying dynamics. However, a consistently low accuracy indicates that there is either a very subtle or no relationship between inputs and output, because neural networks can approximate complex, nonlinear functions (a particular training run may not converge, nevertheless).

This chapter focuses on explaining high target values for simplicity, but this does not limit the range of the approach. If analysts are interested in low- or mid-value cases, y could be transformed accordingly to fit the definition (e.g., $\hat{y} = -y$ for low-value cases).

7.3.1 Basic Architecture

Figure 7.1 depicts the architecture of the model that is based on a fully-connected, feed-forward neural network with one hidden layer and one output node. To compute the (scalar) output h_i of a hidden node i in an artificial neural network, the dot product of the inputs $\boldsymbol{x}$ with the trained weights of the node $\boldsymbol{w^i}$ plus a constant offset b^i (the bias) is fed into a nonlinear, monotonically increasing activation function $\sigma(z)$:

$$h_i(\boldsymbol{x}) = \sigma(\boldsymbol{w^i} \cdot \boldsymbol{x} + b^i) \tag{7.1}$$

The approach uses the sigmoid function $\sigma(z) = \frac{1}{1+e^{-z}}$ as activation function which returns values between zero and one. If the function output is close to

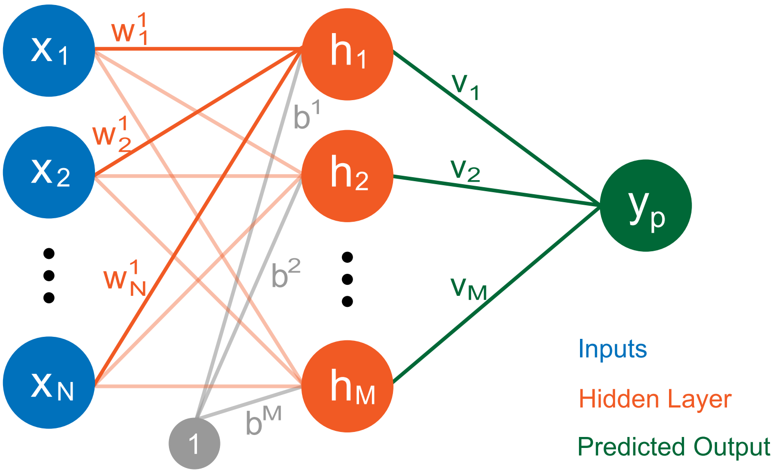


Figure 7.1 — Architecture of the model, with inputs $\boldsymbol{x}$, input-to-node weights $\boldsymbol{w^i}$, biases b^i, hidden node outputs h_i, node-to-output weights v_i, and prediction y_p of the target.

zero, we say that the respective hidden node is *inactive*, otherwise, it is *active* (to a certain degree).

Similar to linear regressions, inputs with positive associated weights (or coefficients) have a positive monotonic relationship with the output of the hidden node (i.e., higher values lead to equal or higher outputs), inputs with negative weights have a negative relationship, and inputs with weights close to zero do not significantly impact the output.

However, artificial neural networks are more powerful than linear regressions, because they use nonlinear activation functions. In simple terms, the activation function enables thresholds, that is, the node's output may only start to grow significantly above zero if the weighted sum reaches a certain value, and it nearly stops to grow once it is close to one.

Apart from this non-linearity, the relation of the hidden node output to its inputs as defined by the weights is still monotonic and smooth, which this approach exploits. Understanding the behavior of individual hidden nodes is therefore easier than understanding the model as a whole. The final output $y_p(\boldsymbol{x})$ is assembled from the outputs of the hidden nodes as weighted sum:

$$y_p(\boldsymbol{x}) = \sum_i h_i(\boldsymbol{x}) v_i \tag{7.2}$$

In the following, all hidden nodes with a positive associated weight v in the final layer are called *positive nodes* (they drive the final prediction up), and nodes with a negative v in the final layer are called *negative nodes*. In contrast to a typical feed-forward architecture, the model architecture does not use the bias in the final layer. This makes sure that at least one positive node has to be active for high target values. Every variable (inputs and output) is scaled so that they are in the range $[0, 1]$. The lack of bias and the normalization of the output ensures that the resulting prediction of the neural network is (only) high if at least one positive node is sufficiently active and negative nodes are mainly inactive. Hence, positive nodes may offer hints which variables and conditions correlate with high target values.

7.3.2 Hidden Node Filtering

Each positive node acts as a filter. The approach assumes that, during training, some hidden nodes capture specific cases that lead to a high target value (*high-target cases*). The architecture requires that at least one positive node has to output a positive value above zero whenever the target value is high, so filtering the data according to which hidden node is active can offer insights into the cases we are interested in. This decomposition enables analysts to study the relationships between input variables and target separately for distinct conditions. In particular, the input weights of a node indicate how important each input is for the computation of this node, and analysts can focus on the most promising variables even if the total number of inputs is rather high. Figure 7.2 visualizes this concept.

While at least one positive node has to output a positive value for a high target value, this is a necessary and not sufficient condition. That is, it does not hold that an active (i.e., its output $h_i(\boldsymbol{x})$ is high) positive node necessarily leads to a high target value prediction. For the analysis goal, it is therefore often not enough to filter the data items solely based on whether a particular hidden node is active.

Let us look at an example to illustrate this. Say, for instance, the target variable peaks if the (single) input is in the middle at 0.5, but it is low if the input is near zero or one. Thus, the visualization of this case should convey that input values around 0.5 lead to high target values. However, one hidden node alone cannot model this case, because, due to the monotonic nature of the activation function, the output of the node cannot go down again after the input has reached 0.5. The neural network could use a second *negative* node to approximate this function. That is, the positive node models the ramp-up to 0.5 and the negative node the ramp-down to zero again. We can say that the second, negative node *inhibits* the output of the positive node if the input is above 0.5. Therefore, we have to take into account the effects of such *inhibitors*

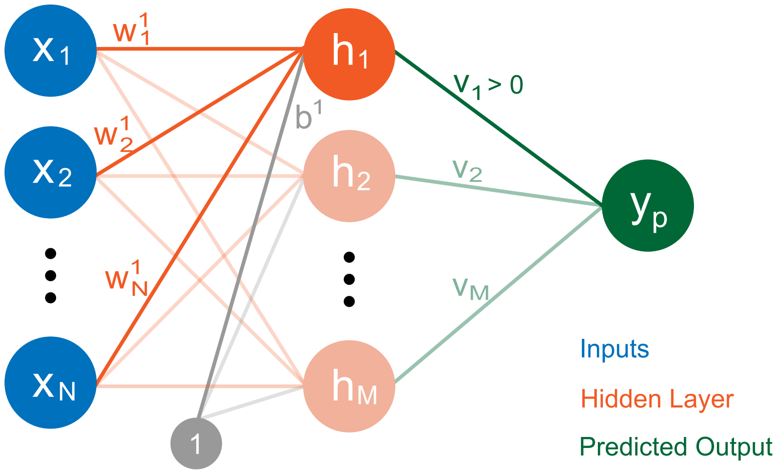


Figure 7.2 — The prediction y_p can only be high if at least one hidden node k with $v_k > 0$ is active (i.e., $h_k \gg 0$) because the hidden node outputs are never negative and the prediction is a weighted sum from these outputs. The analysis of such positive nodes can help to explain high-value cases. For instance, let us assume that $v_1 > 0$ and that the weighted output $v_1 \cdot h_1$ is significantly higher than any other weighed output for a specific set of data items S. Then, the corresponding input weights w_j^1 of said first hidden node can help to explain the characteristics of the items in S since the weighted input sum $\sum_j w_j^1 \cdot x_j$ has to be $\gg 0$. We can derive, for instance, particularly relevant variables j for S based on the magnitude of the corresponding weights $|w_j^1|$.

while a particular hidden node is active. To achieve this, we visualize the data distributions of the inputs and the target output for such data items where the respective hidden node of interest is *contributing* and not just active. This is a smaller subset that only covers cases in which the node is active and also contributing to a higher target value prediction. In the example, this would mean that the node only contributes for input values around 0.5, which is the desired output for the visualization.

Regarding a data item, we say a node is *contributing* if this node is active, *and* the weighted output of the node is high compared to all other weighted outputs of positive nodes, *and* the prediction is high. In other words, we want to detect *salient* stimulations of hidden nodes by the input data that travel through to the final output value. More formally, let V be the set of hidden nodes, then the contribution $c^i(\boldsymbol{x})$ of hidden node i regarding data item $\boldsymbol{x}$ is:

$$c^i(\boldsymbol{x}) = \frac{1}{Z} h_i(\boldsymbol{x}) \frac{\min(y_p(\boldsymbol{x}), h_i(\boldsymbol{x}) v_i)}{\sum_j h_j(\boldsymbol{x}) v_j}, \quad \forall j \in V : v_j > 0 \tag{7.3}$$

We take the final prediction if it is lower than the weighted output of the node to ensure that the node is not suppressed by other (negative) nodes. Z is a scaling factor to retrieve a contribution of 1 for data items that stimulate the particular node the most. The advantage of filtering by contributions instead of just activations is that the resulting 'clusters' can model highly nonlinear relationships. For instance, one cluster could model the case that our target variable correlates linearly with a *if* variables b and c are both within a specific range.

7.3.3 Ranking of Variables

The filtering based on the nodes (representing a soft clustering of the data set) helps analysts to focus on specific parts of the data set, but without a suitable ranking of the variables, it may still be tedious to gain insights into such clusters, particularly if the total number of variables is high. The approach ranks these variables according to their impact on the contribution of a particular hidden node. As explained in Section 7.3.1, the output of a hidden node is monotonically related to the weighted sum of all inputs. On the one hand, this means that the magnitude of the weight is an indicator of the impact of the corresponding variable, which can be easily extracted from the trained model. On the other hand, we also have to take the distribution of the variable into account. A high positive weight is less important if the average value of the variable is close to zero, for instance.

For an input k of node i with a positive correlation regarding node's output (i.e., $w_k^i > 0$), we calculate the average value of the input k whenever the node i is contributing and multiply it with the weight w_k^i. This determines which variable, on average, has a high share of the weighted sum while the node is contributing, making it a driving force for high activations

For an input l of node i with a negative correlation (i.e., $w_l^i < 0$), we determine the average value of the input l whenever the node i is *not* contributing and multiply it with the absolute value of the weight $|w_l^i|$. The resulting value tells us which variable has the highest impact on inhibiting the output whenever the node is not contributing, and is therefore enabling high activations if the values are low.

More formally, let X be the input rows of the data set. The rank r_k^i of the variable k regarding hidden node i is defined as follows (using the contribution defined in Equation 7.3):

$$
\begin{aligned}
r_k^i &= \frac{|w_k^i|}{S} \sum_{x \in X} x_k \, \hat{c}_k^i(\boldsymbol{x}), \quad S = \sum_{x \in X} \hat{c}_k^i(\boldsymbol{x}) \\
\hat{c}_k^i(\boldsymbol{x}) &= \begin{cases} c^i(\boldsymbol{x}), & \text{if } w_k^i > 0 \\ 1 - c^i(\boldsymbol{x}), & \text{otherwise} \end{cases}
\end{aligned}
\tag{7.4}
$$

7.3.4 Homogeneous Regularization

The aim of the approach is to leverage machine learning for decomposing the structure of the target variable into separate components depending on the input variables to explain in which cases the target variable is high. Sometimes, the resulting nodes still combine several cases that should be represented separately, so that analysts can more easily interpret the relations visually. To encourage this disentanglement, a regularization technique for an increased visual interpretability is introduced that encourages positive hidden nodes to be mainly active in high-value predictions, and negative nodes (inhibitors) to be mainly active in low-value predictions. That is, the behavior of individual nodes with respect to their activation pattern should be more *homogeneous*.

This chapter defines a hidden node i as being *positive/negative* if the associated weight v_i to calculate the final prediction is positive/negative. Let τ be the (user-defined) threshold that defines the border between low- and high-value cases of the target variable. For each data item $\boldsymbol{x}$ and hidden node i in the batch, we define:

$$
a_i(\boldsymbol{x}, v_i) = \begin{cases} 1, & \text{if } \; y(\boldsymbol{x}) \geq \tau \wedge v_i < 0 \;\; \vee \;\; y(\boldsymbol{x}) < \tau \wedge v_i > 0 \\ 0, & \text{otherwise} \end{cases}
\tag{7.5}
$$

In other words, $a_i(\boldsymbol{x}, v_i)$ is 1 if the current node i is a positive node and the current target value is below the threshold, or it is 1 if the current node is an inhibitor and the current target value is above the threshold.

Let V be the set of hidden nodes. The loss function L for a single data item $(\boldsymbol{x}, y)$ with the weights W and biases B is then defined as

$$
L(\boldsymbol{x}, y, W, B) = \frac{1}{2}\left(\left(\sum_{i \in V} h_i(\boldsymbol{x}) v_i\right) - y(\boldsymbol{x})\right)^2 + \frac{1}{2}\beta \sum_{i \in V} a_i(\boldsymbol{x}, v_i) h_i(\boldsymbol{x})^2
\tag{7.6}
$$

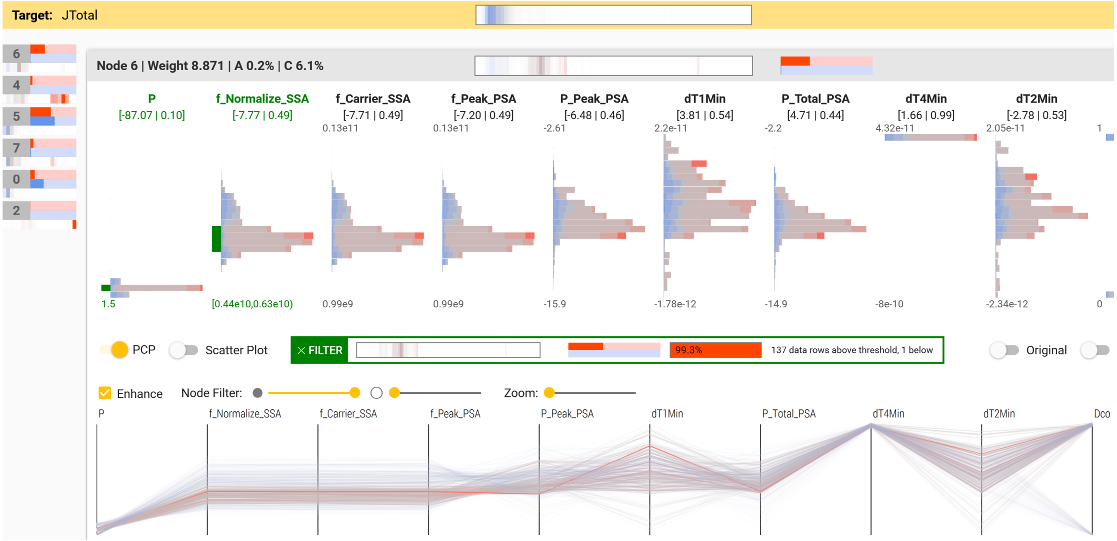
The first part of the loss function is just the typical squared loss to let the network learn accurate predictions. The second part is the homogeneous regularization term that penalizes high node activations of positive nodes for data items that ultimately result in low final outputs, and high node activations of negative nodes for data items that result in high final outputs. The hyper-parameter β defines the strength of the regularization.

As stated in Section 2.5, the goal of any regularization is to train a model that is better according to some criteria (e.g., generalizability to unseen data) by limiting the solution space. In this case, the neural network should be encouraged to model the task in a 'simpler' way such that the visualized behavior of individual nodes is easier to comprehend.

For illustrative purposes, let us assume we have two input variables A and B, and the target variable Y is high if (and only if) *one* of the two input variables is high and the other *low*. A neural network could model the target function as follows: one hidden node is active if the sum $A + B$ is sufficiently high, and the other hidden node with a negative weight is active if the sum $A + B$ is *higher* than the maximum possible value of either input. In this case, the first node captures *both* high-value scenarios (A is high and B is low, as well as B is low and A is high). However, the first node is *also* active in a low-value case, namely if A and B are both high. The final prediction is only low in this case because the second node with the negative weight inhibits the output of the first node if A and B are both high. While such a network makes perfectly fine predictions, it is more difficult to interpret visually, because one node captures both high-value scenarios. The proposed regularization, though, avoids this constellation. The network is encouraged to model the relationship in a different way with a greater specialization of each node. One hidden node would *only* be active if A is high and B low, and the other if B is high and A low, because this leads to a lower loss of the regularization term (while still predicting equally well). The visual representation of the resulting model can then easily explain the two conditions separately.

Section 7.5.2 reports on the results of a hyper-parameter analysis on a synthetic data set with different values for β and several network sizes. They show that the homogeneous regularization ($\beta \geq 0.1$) helps to extract all high-values cases, particularly the more subtle ones. A higher number of hidden nodes also increased the likelihood of detecting relevant patterns. However, the results also show that the approach is generally not very sensitive regarding the hyper-parameters.

The network is trained with mini-batch gradient descent and RMSprop to iteratively derive the weights and biases (see Section 2.1.4). After training, we compute for all data items the respective contribution of each hidden node and rank the variables for each node. We use the contributions of a particular node



Target: JTotal

Node 6 | Weight 8.871 | A 0.2% | C 6.1%

137 data rows above threshold, 1 below

Figure 7.3 — Visual Neural Decomposition of a chip testing measurement data set with the goal to identify cases in which the target variable (here: jitter) exhibits high values. Each node visualizes parts of the data set depending on its activation.

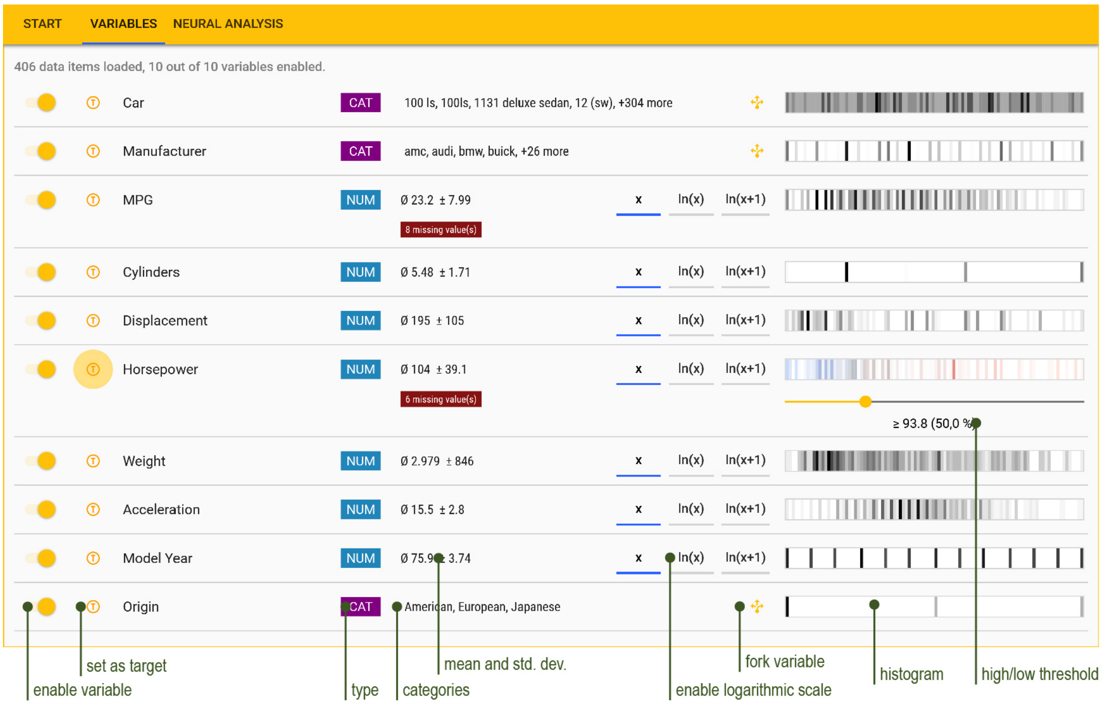
to filter the data items for the histogram and parallel coordinate plots that are described in detail in Section 7.4.

7.4 Application Design

The proposed system extracts and visually explains multivariable correlations in a data set regarding a specified target variable. After performing the neural decomposition as described in the previous section, it visualizes the distributions of the input variables in relation to the target variable separately for each positive hidden node in the model, based on the calculated contributions (Section 7.3.2). This helps analysts to understand which variables and ranges correlate with high target values, as each cluster may represent a different group of conditions.

It should be noted that *target variable* always refers to the actual variable and values in the data set, not to the predictions of the model. The predictions are only used to derive the soft clustering (Section 7.3.2).

The system provides an integrated workflow that enables analysts to load data sets, configure variables, set parameters, train the model, visualize the results, and interact with the views, all within one application. A typical workflow starts with selecting the target variable of the loaded data set. The analyst



START VARIABLES NEURAL ANALYSIS

406 data items loaded, 10 out of 10 variables enabled.

Car	CAT	100 ls, 100ls, 1131 deluxe sedan, 12 (sw), +304 more				
Manufacturer	CAT	amc, audi, bmw, buick, +26 more				
MPG	NUM	Ø 23.2 ± 7.99 8 missing value(s)	x	ln(x)	ln(x+1)	
Cylinders	NUM	Ø 5.48 ± 1.71	x	ln(x)	ln(x+1)	
Displacement	NUM	Ø 195 ± 105	x	ln(x)	ln(x+1)	
Horsepower	NUM	Ø 104 ± 39.1 6 missing value(s)	x	ln(x)	ln(x+1)	≥ 93.8 (50,0 %
Weight	NUM	Ø 2.979 ± 846	x	ln(x)	ln(x+1)	
Acceleration	NUM	Ø 15.5 ± 2.8	x	ln(x)	ln(x+1)	
Model Year	NUM	Ø 75.9 ± 3.74	x	ln(x)	ln(x+1)	
Origin	CAT	American, European, Japanese				

Figure 7.4 — All available variables of the automobiles data set with previews of the data, scale, and histogram.

can then train a neural network that tries to predict the target value based on the other variables. After training, the user interface visualizes the resulting clustering with stacked histograms for each hidden node. They offer a compact visual representation of the conditions that lead to high target values. The histograms show the distribution of the variables for a subset of data items that stimulate the particular node. Analysts can then explore and verify the found correlations in a targeted manner with interactive, node-specific parallel coordinate plots and by using simple range filters.

7.4.1 Variables

After loading the data set, the detected variables are listed in the respective tab as shown in Figure 7.4. Here, the 1983 Data Exposition data set (Ramos and Donoho [1983]) has been loaded that contains a list of (rather ancient) automobiles with several properties and technical specifications, for instance, the horsepower of the car and its driving range in miles per gallon. Analysts can choose which variables should be part of the subsequent analysis process with the toggles on the very left of each row. To select the target variable, they can click on the *(T)* button next to the name of the respective variable. In

Figure 7.4, *Horsepower* was selected as target variable, that is, the analyst wants to explore how the other variables relate to the horsepower of the car. One insight could be that a fast acceleration time correlates with a powerful engine, for instance.

Upon loading, the application automatically analyzes the given data set to determine the type of each variable (categorical or numerical) and whether a logarithmic scale should be applied. Each category is encoded as a number between zero and one. While this embedding allows to represent hundreds of categories with just one variable, in machine learning it is often beneficial to split categorical variables into distinct binary variables that represent whether the specific category is *on*, that is, whether it is the current value of the original variable. This is also necessary if users want to designate a specific category as target. Analysts can click on the yellow *fork* button left to the histogram (in Figure 7.4) to generate said distinct variables for each category.

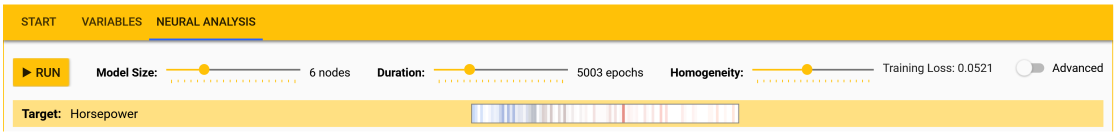
Numeric inputs are scaled to the range from zero to one, after having transformed them to the logarithm if needed. Missing values are replaced with the respective regular minimum (0 after scaling) to avoid distortions.

The middle of each row displays the type of the variable alongside statistical properties such as the number of categories or mean and standard deviation. Next to it, analysts can override the automatic decision whether a logarithmic scale should be applied or not.

To the right, a histogram provides insights into the distribution of the variable. A darker shade indicates a higher number of values around that region. For the selected target variable, a slider underneath the histogram appears which allows analysts to define a custom threshold. The percentage in brackets indicates how many data rows fit this threshold and count as high-value cases. The goal of the approach is to visually explain in which conditions the target variable is higher than said threshold. The middle value between maximum and minimum of the respective variable is the default, but in Figure 7.4, the analyst has changed it to the median.

7.4.2 Model Training

In the third tab (*Neural Analysis*), analysts can start the training of the model. Figure 7.5 shows how analysts can modify the model size, duration, and the strength of the regularization with sliders located at the top. Bigger models may better capture small-sized effects in the data, but take longer to train and can lead to some redundant nodes showing mostly the same information. The resulting error on the training set is depicted at the top-right corner after each run. Underneath the sliders, the interface shows again the histogram of the



Model Size: 6 nodes

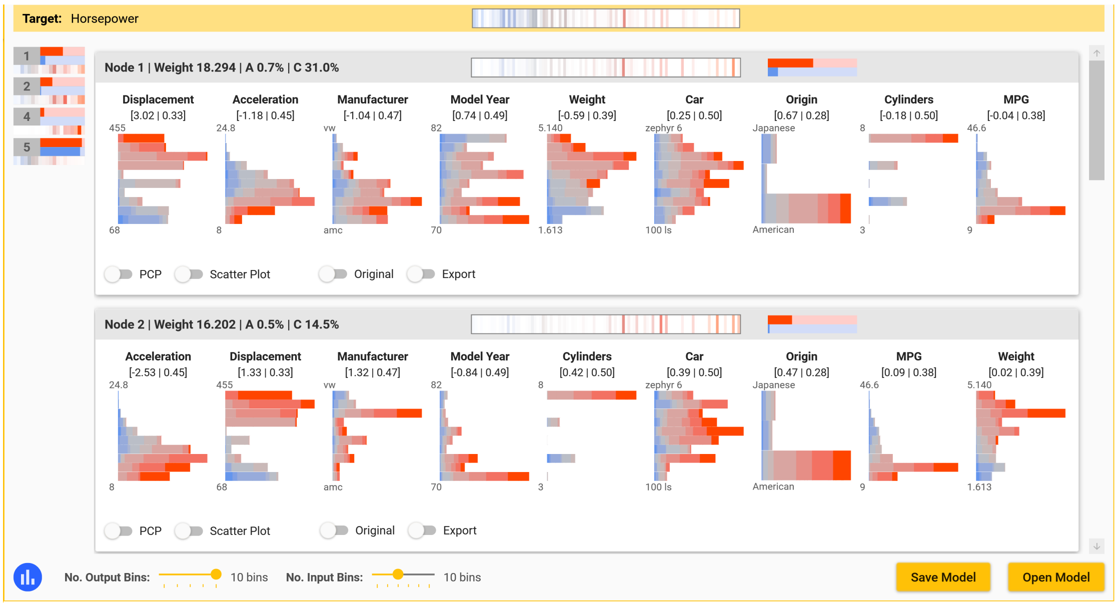
Duration: 5003 epochs

Homogeneity:

Training Loss: 0.0521

Target: Horsepower

Figure 7.5 — Meta-parameters of the neural network training, e.g., complexity of the model in terms of the number of hidden nodes, or duration in terms of the number of iterations.


Target: Horsepower

Node 1 | Weight 18.294 | A 0.7% | C 31.0%

Displacement [3.02 | 0.33] 455 68
Acceleration [-1.18 | 0.45] 24.8 8
Manufacturer [-1.04 | 0.47] vw amc
Model Year [0.74 | 0.49] 82 70
Weight [-0.59 | 0.39] 5.140 1.613
Car [0.25 | 0.50] zephyr 6 100 ls
Origin [0.67 | 0.28] Japanese American
Cylinders [-0.18 | 0.50] 8 3
MPG [-0.04 | 0.38] 46.6 9
PCP
Scatter Plot
Original
Export

Node 2 | Weight 16.202 | A 0.5% | C 14.5%

Acceleration [-2.53 | 0.45] 24.8 8
Displacement [1.33 | 0.33] 455 68
Manufacturer [1.32 | 0.47] vw amc
Model Year [-0.84 | 0.49] 82 70
Cylinders [0.42 | 0.50] 8 3
Car [0.39 | 0.50] zephyr 6 100 ls
Origin [0.47 | 0.28] Japanese American
MPG [0.09 | 0.38] 46.6 9
Weight [0.02 | 0.39] 5.140 1.613
PCP
Scatter Plot
Original
Export
1
2
4
5
Save Model
Open Model

No. Output Bins: 10 bins

No. Input Bins: 10 bins

Figure 7.6 — Visualization of the resulting model. Each card represents one hidden node and visualizes the data set filtered by the contribution of the respective node. A compact overview of the nodes is shown on the left.

selected target variable, which is helpful for comparison with the subsequent node visualizations.

7.4.3 Node Visualization

The system enables analysts to investigate under which conditions a specific target variable adopts values above a certain threshold (*high-value cases*). Data items with a target value above the threshold are called *high-value items*. The approach assumes that some of the positive hidden nodes in the model specialize in particular high-value cases that can then be explained visually. The distinction between high- and low-value items and cases should improve the utility of the visualization, and it also influences the homogeneous regularization (Section 7.3.4). For instance, the interface generally maps high-value items

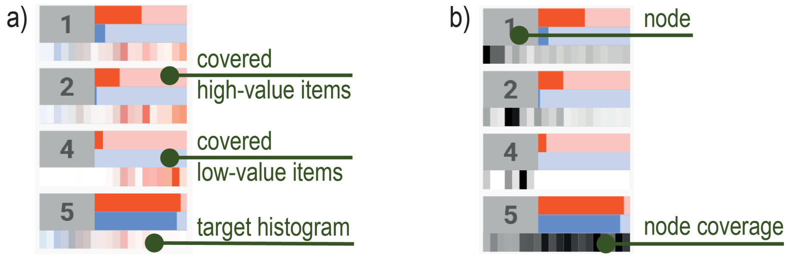


Figure 7.7 — Compact overview of all positive nodes: a) node-specific histogram of target, b) node-specific coverage of high-value cases

to red colors, and low-value items to blue ones. However, the neural network learns to predict the actual target value irrespectively of the threshold.

Figure 7.6 shows the resulting visualizations of the positive nodes after training. For each node (and thus, cluster), a *card* shows the distribution of the data subset the respective node contributes to. The card-based layout maps well to the neural decomposition, in that each node represents one distinct cluster of high-value cases that analysts can explore separately, with aggregated statistics for each case. At the same time, it also enables the comparison between nodes by presenting the cards in a list, with the possibility to rank nodes by their relevancy for the problem at hand.

Each card presents stacked histograms (Section 7.4.4) of the input variables, ordered by the importance of the respective variable for the cluster as described in Section 7.3.3. They offer a visual summary for which input ranges the node is contributing in relation to the target values. Variables with a very low importance score are not shown to improve the clarity of the visualization (5% of the first, most important variable per default). The actual coefficient (weight) and average (normalized) value of each input is displayed in squared brackets below the name of the variable. The gray header displays the weight, average activation (A), and average contribution (C) of the node. The red and blue bars to the right indicate how many of the high- and low-value items the node covers. For instance, if the red bar is at 100%, this means that the node is contributing to all data items with a target value above the threshold. Analogously, the blue bar represents the proportion of the low-value cases where the node is contributing. In the ideal case, the blue bars should be rather small. If red and blue bars are both strong, the respective cluster covers low- and high-value cases to a similar extent, and thus does not help analysts in understanding which input values correlate with high-value items.

Figure 7.6 reveals that *Displacement* has the most impact on Node 1 with a weight of 3.026, followed by *Acceleration*. The signs of the weights indicate a positive correlation for the variable *Weight* (bigger cars in the data set seem to have more horsepower) and a negative correlation with *Acceleration* (faster acceleration times seem to coincide with more horsepower). It should be noted that these node-specific correlations do not necessarily mean that there is an overall statistical correlation with the target variable.

Target Histogram and Node Coverage

In the middle of the gray header, a small histogram depicts the distribution of the target value whenever the node is contributing (target histogram). Users can switch to display the black *node coverage* instead of the target histogram using the blue toggle button on the bottom left of the window (Figure 7.6). While the target histogram visualizes on which regions of the *target variable* the node specializes, the node coverage instead shows for which high-value *data items* the node is contributing. This helps to understand which nodes complement one another and which cover similar data items.

For instance, Figure 7.7 a) shows that Node 4 only contributes to high-value items (target histogram only shows red densities), whereas Nodes 1 and 2 also slightly contribute to a few low-value items. The target histograms of Node 1 and 2 are similar, so the nodes cover items with a similar distribution of the target variable, but the node coverage (Figure 7.7 b)) reveals that Node 2 contributes most to different data items than Node 1 as the black regions do not overlap much.

To compute the node coverage, we take each high-value item, sort them by which node contributes most on the respective item, assign each a running number, and then build a histogram of all the respective running numbers where the particular node is contributing, that is, each data item is represented by a vertical line colored from white to black depending on the contribution.

On the left side of the window (Figure 7.6), the node summary provides analysts an overview of all nodes with compact versions of the histograms and case coverages that are displayed in the respective headers. Depending on the setting, either the target histogram (Figure 7.7 a)) or node coverage (Figure 7.7 b)) is shown.

Node Ranking

The total number of nodes can be high. To help users focus on promising ones, the interface shows those nodes on top which explain high-value cases. Hence, these nodes should contribute mostly whenever the target value is high. The total number of affected high-value items should also be taken into account to

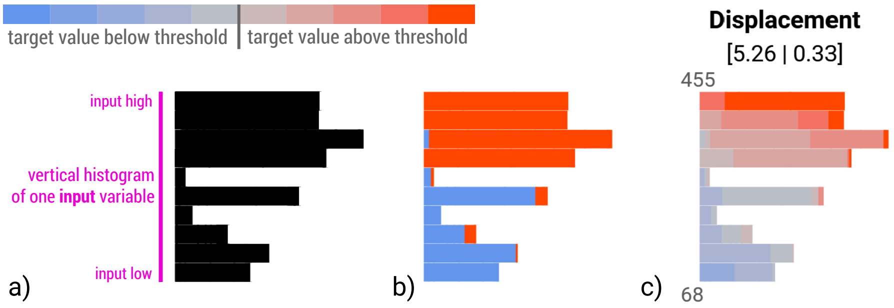


Figure 7.8 — Histogram of one particular input variable of a node: a) plain, b) two stacked bars per input bin for high- (red) and low-value (blue) cases, c) ten stacked bars per bin to visualize distribution of the target variable based on the input variable.

rank bigger cases higher. This ranking is computed by multiplying the number of affected high-value items of the node with the logarithm of the total number of low-value items divided by the number of affected low-value items. Taking the logarithm of the fraction ensures that nodes covering many high-value items but *relatively* few low-value cases are boosted.

7.4.4 Stacked Histograms

The interface should visualize for which kind of input ranges the particular hidden node is mostly contributing. To achieve this, we generate node-specific histograms of the input variables. Given an input variable, we divide its range into equally distributed bins and count how many items in the data set fall within each bin, *weighted* by the contribution of the node. This results in a histogram of the variable that only incorporates data items where the node is contributing. This means that if the node is not stimulated by a particular data item then this item is not part of the histogram. Figure 7.8 a) shows how the resulting vertical histogram could look like.

This histogram shows the distribution of the input variable whenever the node is contributing, but it would further help if this could be combined with the distribution of the target value. That is, given a specific range of the input variable, it would help to visualize the proportion of high-value items versus low-value items whenever the node is contributing. Hence, we count high- and low-value data items separately for each bin and display the resulting histogram as a stacked bar chart.

In other words, the system builds two separate histograms, one for those data items in the cluster with a high target value (red bars), and one for those with

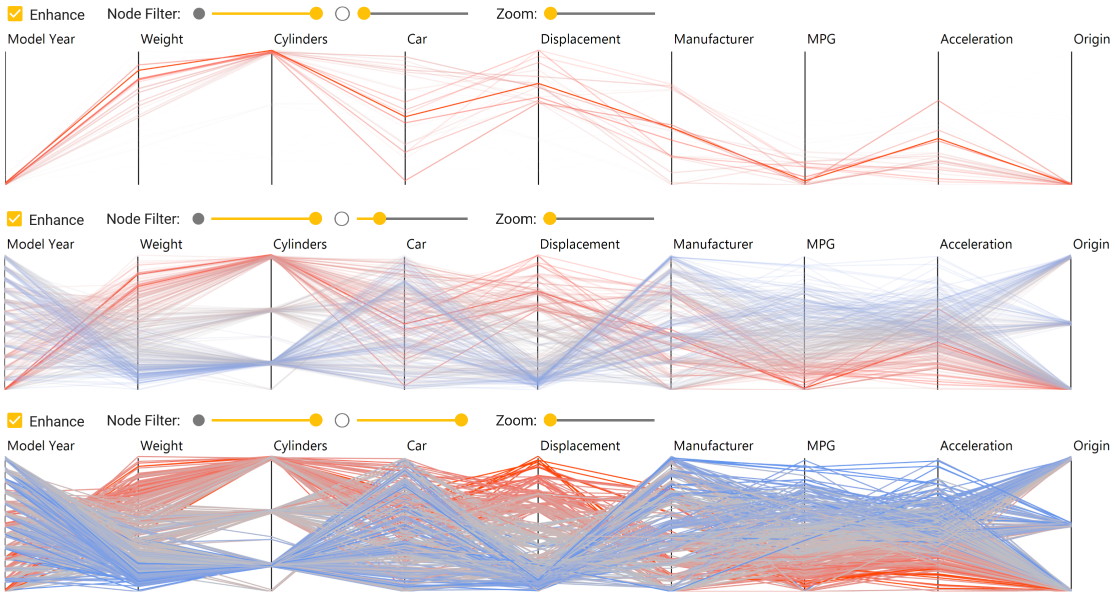


Figure 7.9 — Node-specific PCPs with different filter settings. The columns are ordered by the importance of the inputs for the node. Top: Plot using only data items where this node is contributing. Middle: Plot includes remaining data items, but diminished. Bottom: Plot of all data items.

a low target value (blue bars). For instance, Figure 7.8 b) shows that for every data item where the particular node is contributing *and* where the *Displacement* value falls into the top bin, the target value is always above the threshold (the bar is completely red). Conversely, low *Displacement* values are associated with target values below the threshold, because the lower bars are nearly completely blue.

Instead of just distinguishing two cases (above vs. below the threshold), the interface enables users to increase the granularity to multiple bins. This leads to histograms in a histogram. For each bin of the input variable (rows), it visualizes the distribution of the target variable (stacked bars within the row).

Analysts can choose the granularity of the histograms with two sliders at the bottom of the window (Figure 7.6). Figure 7.8 c) shows the resulting visualization with ten input bins and ten target bins using colors from blue to red. Within the respective cluster, *Displacement* has a near-linear, positive correlation with the target value (horsepower), because, starting from the bottom, the bars change from blue-ish over gray to red-ish.

7.4.5 Parallel Coordinate Plot

The interface employs node-specific parallel coordinate plots (PCPs) to let analysts explore *parts* of the data set at a time and investigate relationships between the variables. While the stacked histograms help to summarize which variables and ranges correlate with the target, the PCPs allow for a detailed analysis how *exactly* the input variables relate to each other for the given case represented by the node. For instance, Node 1 in Figure 7.6 indicates that cars in the data set with high displacement and fast acceleration time have high horsepower. The PCP shows that high displacement generally relates to low acceleration times, but, in addition to the stacked histograms, it reveals that some cars with very high displacement only have average acceleration times.

One shortcoming of such plots is that it is often not obvious how the axes should be arranged, even though the order of the columns has a major influence on which patterns and correlations analysts can detect. To tackle this challenge, the interface uses the same order for the axes as for the stacked histograms. That is, the columns are arranged according to their importance for the respective case represented by the node.

Initially, only paths are drawn from data items where the node is contributing (filtered data), but this can be changed with two sliders above the plot, which are called *node filters*. The filtering mitigates occlusion issues that make it difficult to trace lines and recognize relationships for larger data sets. The first slider (filled circle) determines the general opacity of the *filtered data*, and the second (outline of a circle) the opacity of the *remaining data*. If both sliders are at zero, everything is transparent and nothing is shown. If both are at maximum, the complete data set is displayed (but with the node-specific axis order).

Figure 7.9 shows three different filter settings of one node. In all three cases, the first slider is at maximum, but the second slider changes from zero to intermediate to full. This means that the remaining data is slowly faded in, which helps to relate the specific conditions to the entire data set. In this example, the node focuses on heavy American (lowest position of *Origin*) cars in the early 1970s (lowest position of *Model Year*) that exhibit high horsepower (all lines are red). However, if we only looked at the plot of the entire data set this pattern would be hard to detect visually.

7.4.6 Scatter Plot

Analysts can turn on a node-specific scatter plot after selecting two variables by clicking on their mame. Similar to the parallel coordinate plots, two sliders

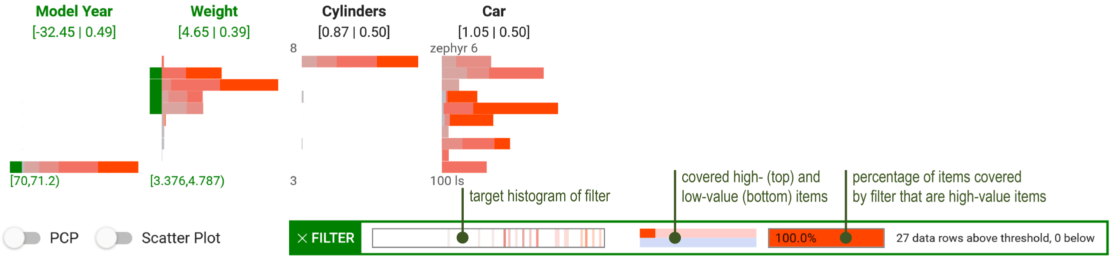


Figure 7.10 — The analyst has selected several bars (green mark on the left) to build a simple range filter. The resulting histogram and statistics are shown below, which in this case indicate that heavier cars in the set built around the 1970s also exhibit above-average horsepower.

allow users to choose whether all data rows should be shown, only those where the node is contributing, or all rows except the ones where the node is contributing. While such scatter plots can only reveal the behavior of two input variables with regard to the target variable, they are still helpful to immediately verify whether two of the most important variables already exhibit a strong relationship with high target values.

7.4.7 Range Filter

Using the stacked histograms, analysts can build range filters to test hypotheses on interesting subsets of data items. These filters reduce the set of data items to the ones having values in the specified ranges.

The stacked histograms in Figure 7.10 indicate that the respective node specializes in ancient, heavy, fast, American cars with many cylinders and high horsepower. To validate whether most of the heavy cars around 1970 in the set have high horsepower, analysts can select the bottom bar of the first variable and the first few bars from the top of the *Weight* variable by clicking on them. Little green rectangles appear to the left of each selected bar, the name of the variable is marked in green, and a summary of the selected ranges (also in green) appears at the bottom of the histogram. The resulting histogram and statistics of the filter appear below in the box with the green border. Here, the range filter selects 27 cars and all exhibit horsepower above the threshold. The small red bar in the middle of the green box (above the blue bar) shows that the filter covers roughly 10% of all high-value cases. Hence, cars in the data set from around 1970 which are heavier than average also have above-average horsepower.

Range filters always use the complete data set and do not depend on any node

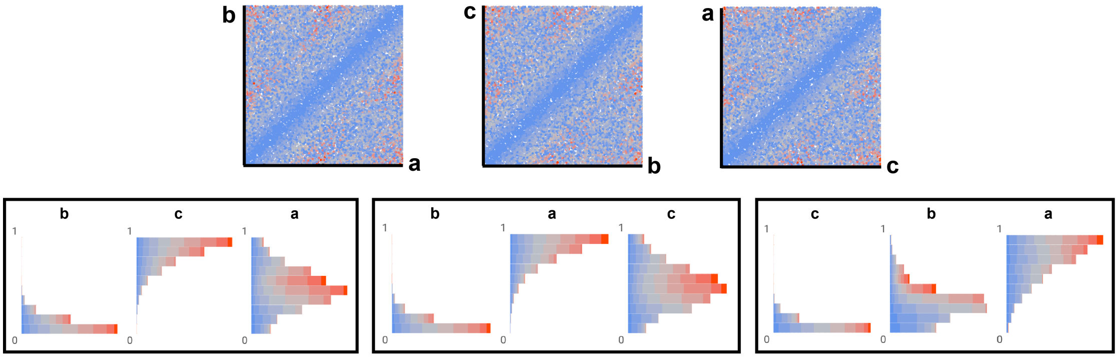


Figure 7.11 — Comparison between scatter plots (top) and the proposed approach (bottom) to visualize a four-dimensional artificial data set containing three different high-value patterns (different variable order due to importance ranking). The target variable is mapped to a color scale from blue to red.

activation. The node-specific visualizations, however, guide analysts to define such range filters.

7.5 Evaluation

This section first shows that the proposed approach can detect and visualize correlations that involve more than two variables. Afterward, the method is applied to real-world data sets to show its utility. Finally, the section reports on qualitative feedback on the suitability of the approach from a domain expert.

7.5.1 Identification of High-Value Cases

The approach was applied to a synthetic data set to test its capability of identifying correlations between more than two variables. The data set contains three input variables a, b, c and 27,000 data rows that correlate with high values of a target variable under the following circumstances:

$$y(a,b,c) = \min(|a-b|, |b-c|, |c-a|) \tag{7.7}$$

The target value reaches its maximum of 0.5 only if one variable is 0, one is 0.5, and the third is 1. The test script uniformly sampled the input space $a, b, c \in [0,1]$ and generated 27,000 rows. The target function $y(a,b,c)$ was designed such that the resulting data set poses several challenges. First, only about 12% of the items have target values above the threshold of 0.25, and only 0.8% above 0.4.

That is, the machine learning model has few high-value samples to learn from. Second, the target highly depends on the combination of all three variables, that is, if one plots individual variables or pair of variables against the target value, one can hardly recognize any correlation. Third, there are three different patterns that exhibit high target values, which would be occluded in a parallel coordinates plot. Hence, this data set is a benchmark for testing the method under challenging conditions.

Figure 7.11 shows the resulting stacked histograms of three hidden nodes of the trained model, and scatter plots of every combination of a, b, c for comparison. The target variable is mapped to a color scale from blue over gray to red in all plots. From the scatter plots at the top, the analyst can conclude that the target is zero if any two variables have equal values, but it is not possible to recognize the three clusters of the ground-truth. Conversely, the stacked histograms at the bottom reveal the three different patterns that lead to high values of y, namely the three permutations of a, b, c where one variable is near 0, one is near 1, and the third is around the middle. For instance, the first plot on the bottom-left indicates high target values (gray- and red-like bars) if b is low, c high, and a around 0.5.

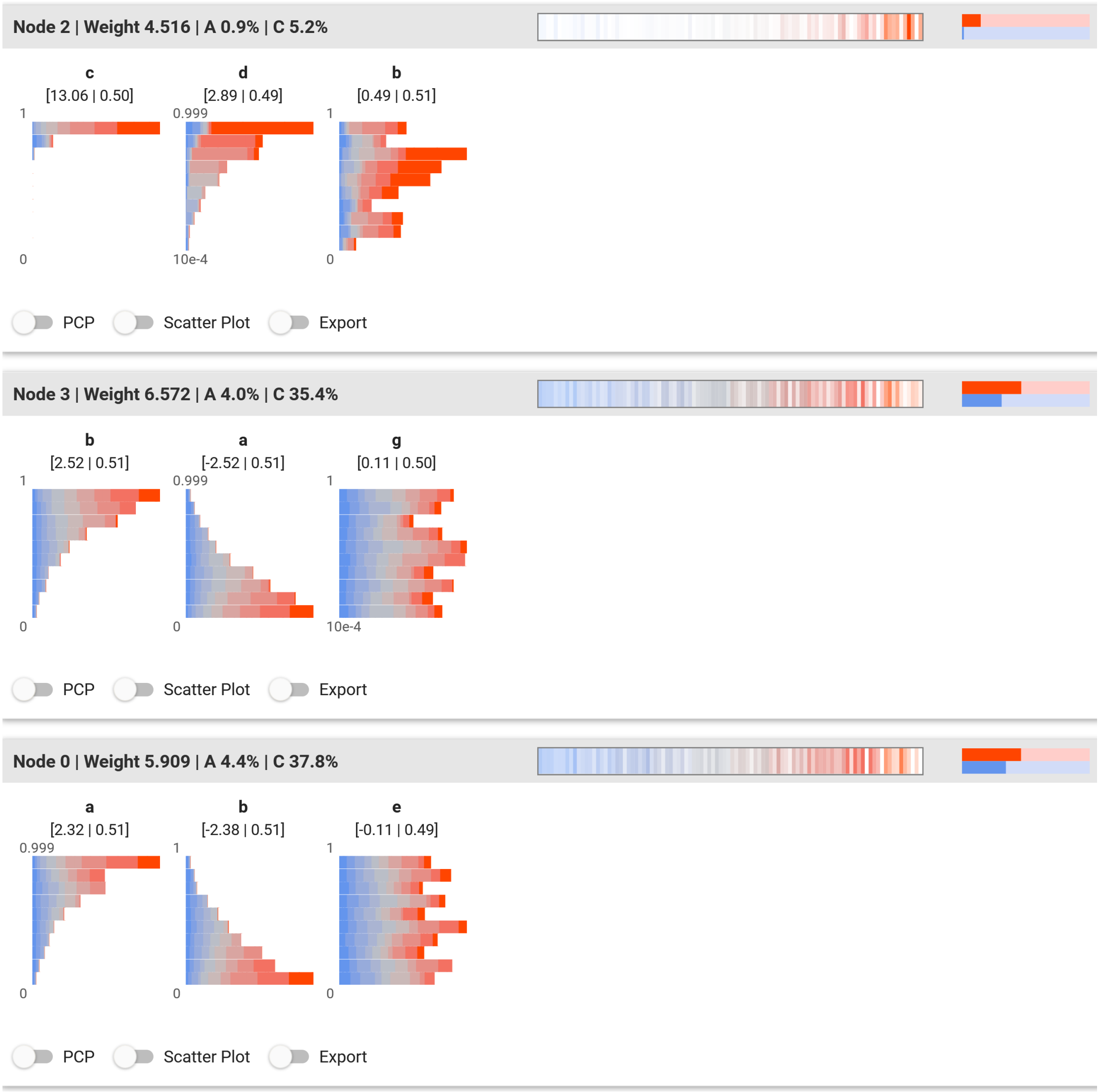
7.5.2 Hyper-Parameter Analysis

The proposed neural network-based model depends mainly on two hyper-parameters: the number of nodes (model complexity) and β that defines the strength of the homogeneous regularization. In typical usage scenarios (with up to a hundred input variables and up to tens of thousands of items), it should be fine to use a model with up to 20 hidden nodes and a value of β that ranges between 0 and 2. In general, if analysts want to extract (or expect) more complex relationships between the variables, they should choose a higher number of nodes (e.g., 10). A more complex model, however, takes longer to train and may lead to an increase of redundant nodes that show similar patterns. The homogeneous regularization ($\beta > 0$) encourages the model to better distribute groups of conditions across several nodes. If β is set very high then the model will not learn to predict the actual target value and analysts will not find interesting insights.

This section reports on the results of a synthetic benchmark that investigates how the hyper-parameters influence the model's capability to detect high-value cases, and how consistent the results are across several runs. The benchmark data set contains 2,000 items with ten input variables (a to j) that were randomly sampled in the range $[0, 1]$. It contains three high-value cases that lead to high values of the target variable y:

Table 7.1 — Results of the hyperparameter analysis on a synthetic data set. Several configurations were tested with varying number of hidden nodes (#Nodes) and strength of the homogeneous regularization (β). For each of the ten runs per configuration, the table depicts whether the approach successfully extracted the high-value cases 1, 2 or 3.

	Case 1 correctly identified (yes / no)	**Case 2** correctly identified (yes / no)	**Case 3** correctly identified (yes / no)
#Nodes = 2			
$\beta = 0$	3 / 7	3 / 7	0 / 10
$\beta = 0.1$	10 / 0	10 / 0	0 / 10
$\beta = 0.5$	10 / 0	10 / 0	0 / 10
$\beta = 2$	10 / 0	10 / 0	0 / 10
#Nodes = 4			
$\beta = 0$	10 / 0	10 / 0	0 / 10
$\beta = 0.1$	10 / 0	10 / 0	3 / 7
$\beta = 0.5$	10 / 0	10 / 0	8 / 2
$\beta = 2$	10 / 0	10 / 0	10 / 0
#Nodes = 6			
$\beta = 0$	10 / 0	10 / 0	3 / 7
$\beta = 0.1$	10 / 0	10 / 0	6 / 4
$\beta = 0.5$	10 / 0	10 / 0	9 / 1
$\beta = 2$	10 / 0	10 / 0	10 / 0
#Nodes = 10			
$\beta = 0$	10 / 0	10 / 0	0 / 10
$\beta = 0.1$	10 / 0	10 / 0	9 / 1
$\beta = 0.5$	10 / 0	10 / 0	10 / 0
$\beta = 2$	10 / 0	10 / 0	10 / 0



Node 2 | Weight 4.516 | A 0.9% | C 5.2%

Node 3 | Weight 6.572 | A 4.0% | C 35.4%

Node 0 | Weight 5.909 | A 4.4% | C 37.8%

Figure 7.12 — Expected output of the approach with the synthetic data set. All three cases are present and interpretable. Here, Node 2 (top) represents case 3, Node 3 (middle) case 2, and Node 0 (bottom) case 1. Per default, variables that influence the node's output by less than 5% (compared to the most important variable in the cluster) are not shown.

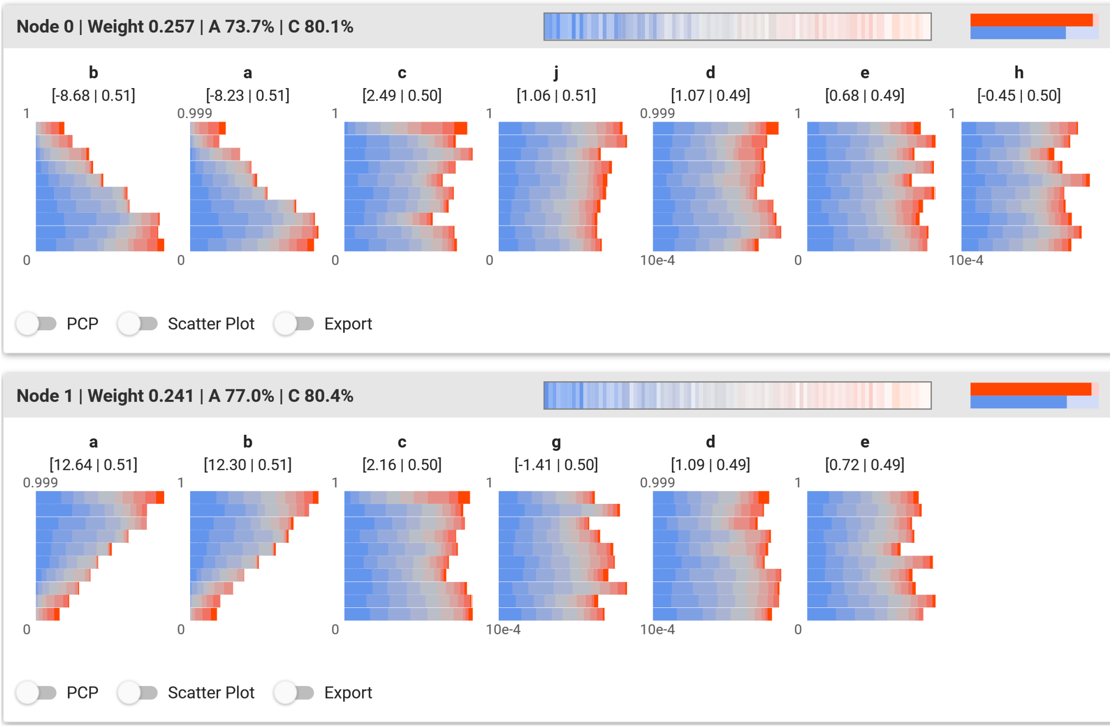


Figure 7.13 — Output of the approach with the synthetic data set when the model fails to capture the relationships in an interpretable way. While the variables a and b are correctly identified as important, the clustering and histograms are difficult to interpret.

1. a is high and b is low,
2. b is high and a is low,
3. if (and only if) c is above 0.9 then d correlates linearly with y

Hence, six variables do not correlate with y and represent noise in the data set. More formally, the ground truth is defined as:

$$y = \max\left(|a - b|, \lambda d\right), \quad \lambda = 1 \text{ if } c > 0.9, \text{ otherwise } 0 \tag{7.8}$$

The first two cases basically represent a continuous relaxation of a XOR b. The ’exclusive or’-problem is harder as it may seem because a linear model or a single perceptron are not able to model this (Minsky and Papert [1969]). The first two cases often overshadow the third, so the third case is challenging to extract. For each hyper-parameter configuration, the method was run ten times and it was determined in how many runs the results included cases 1, 2, or 3.

The mini-batch size was set to 256 and the number of iterations to 10,000. The largest model with ten nodes took about one second to train.

Table 7.1 lists the results of the analysis. In general, the regularization was necessary to extract the third case in a stable manner. With a sufficiently complex model, the approach was always able to extract the first two cases. However, the smallest model with just two nodes often failed to reveal meaningful insights without additional regularization.

Figure 7.12 depicts the expected output where all three high-value cases that are present in the data set are correctly extracted. For instance, Node 2 at the top shows the linear relationship of d with the target given that c is very high. Figure 7.13 depicts how the result can look like if the method failed to visualize any of the high-value cases. Here, it is difficult to derive a meaningful interpretation from the stacked histograms, even though variables a and b have correctly been identified as important. Such a result would count as *no* for all three cases in Table 7.1.

This analysis shows that the proposed homogeneous regularization technique is indeed helpful to retrieve models that can be interpreted visually. Furthermore, it indicates that analysts do not need to find a specific, narrowly defined set of hyper-parameters to extract high-value cases, because in this analysis a range of different hyper-parameter values led to equally meaningful results.

7.5.3 Use Cases on Real-World Data Sets

VND supports the visual analysis of high-value cases in large multivariate data sets across a wide spectrum of use cases where the number of variables can be high. Subsequently, this section presents two real-world use cases from different domains: IC chip testing and population surveys.

Chip Testing Measurements

Detecting and understanding complex input patterns that lead to errors in chip testing is a difficult endeavor. The approach was applied to a data set with measurements of a clock chip from a research project partner, a vendor of automatic test equipment. The goal is to explore which conditions lead to high jitter as indicated by the target variable *JTotal*. Hence, the target variable is continuous, and the higher the value, the worse. The set consists of 2049 data items and 43 variables including the target variable, and about 18% of the items exhibit jitter above the desired threshold.

Figure 7.14 presents the resulting top three nodes after training. These nodes mostly cover different parts of the data set according to the node coverage

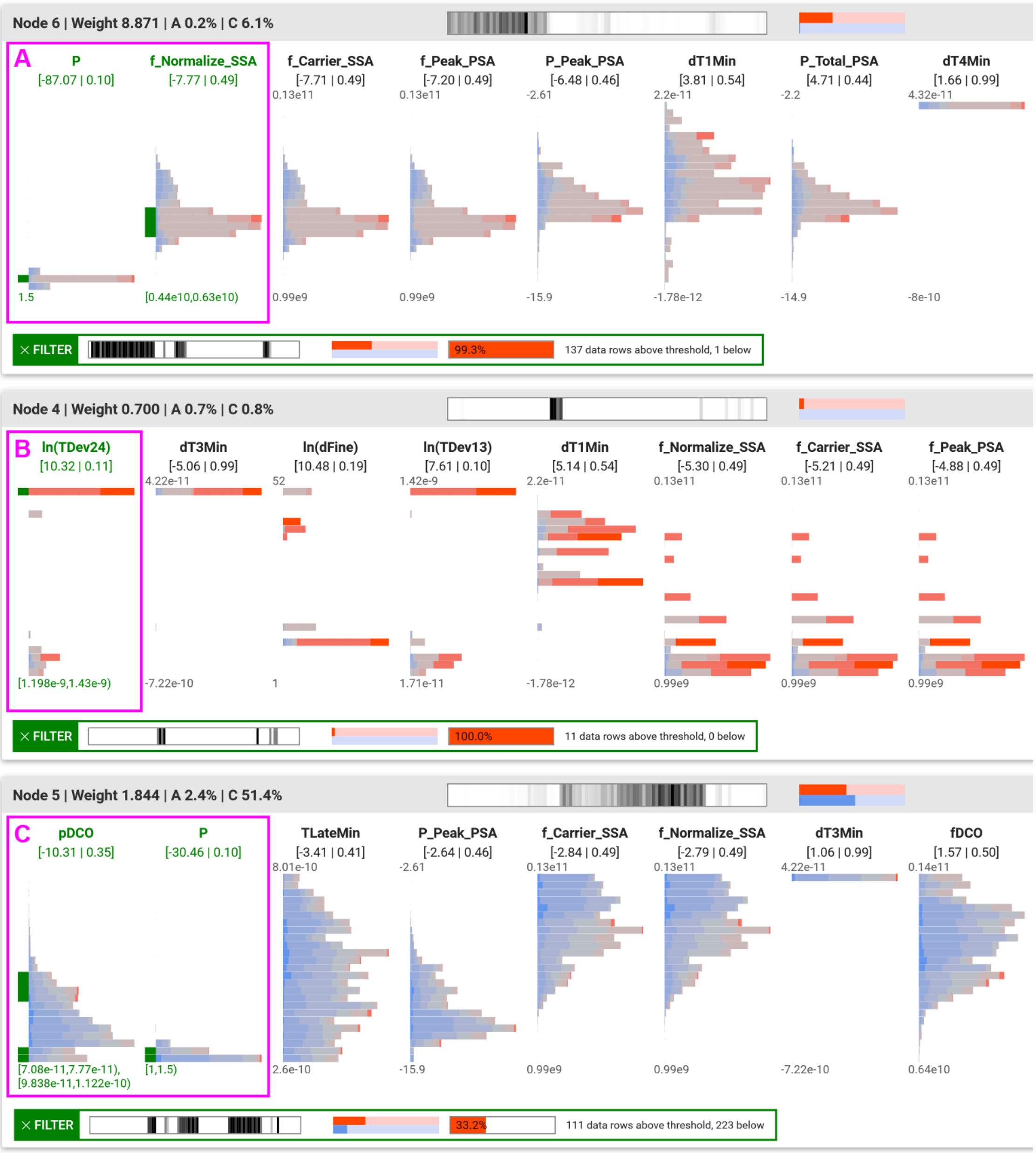


Figure 7.14 — Decomposition of a chip testing data set. Higher target values (gray to red) correspond to faulty behavior of the clock device (jitter).

histograms (in black). The output histograms (Figure 7.3 shows the compact version on the very left) reveal that Node 6 and 5 mostly contribute to lower output values (jitter) around the threshold, whereas Node 4 specializes on few very high-value cases. The Node 6-specific parallel coordinate plot is visible in Figure 7.3 and depicts all data items where Node 6 is contributing. This plot and the applied range filter in Figure 7.14 (A) show that a specific combination of the two most important variables for that first node almost always leads to target values above the threshold and can explain about half of all high-value cases. For Node 4, the first variable is already enough to define 11 items with high jitter (B). However, Node 5 shows a less conclusive picture (C). The applied range filter and the red bar in the header indicate that the pattern represented by the node exhibits above-average jitter, but the node also contributes to several low-value cases. This could mean that these conditions lead to unstable behavior which is sometimes in- and sometimes out-of-spec.

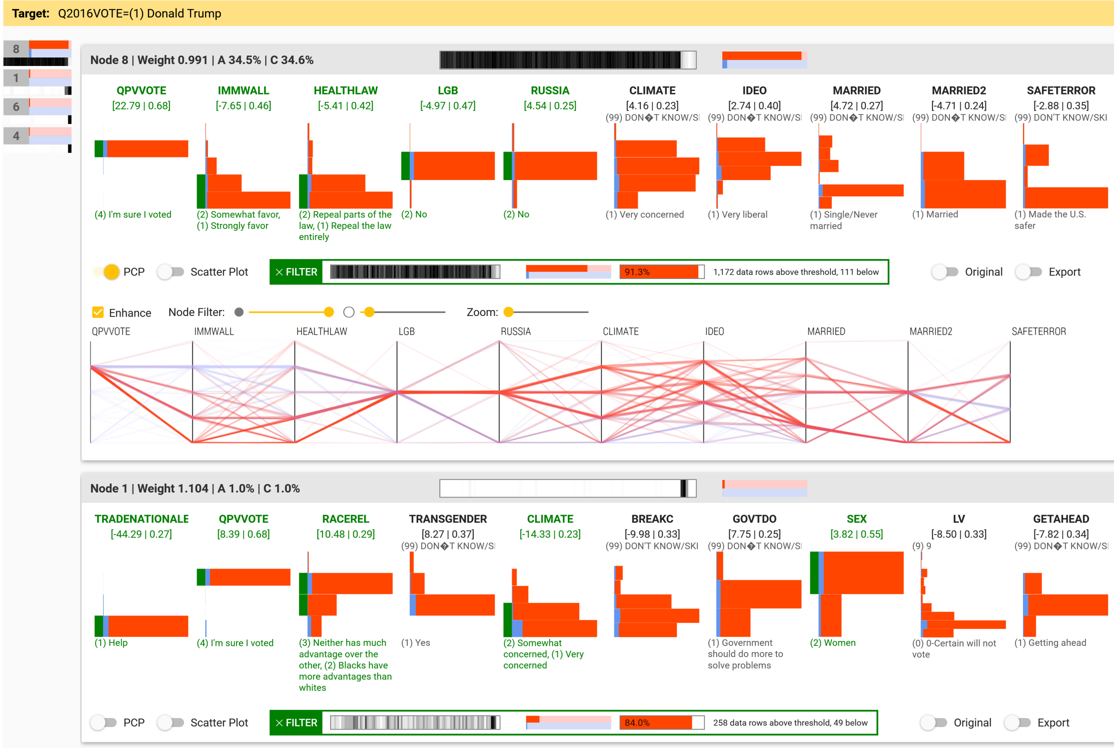
AP VoteCast 2018

The independent social research organization NORC at the University of Chicago conducted a survey of 138,929 registered voters in 2018 for The Associated Press and Fox News (Tompson and Benz [2019]). Each participant was asked a subset of the questions to reduce the duration of each session. In this use case, the analysis goal is to learn more about the voting behavior in the US presidential election in 2016. The analysis was done on all responses in which the participants were asked who they voted for in 2016. The target was either whether the participant answered Donald Trump (1 yes, 0 no), or Hillary Clinton (1 yes, 0 no). The resulting data set consists of 38,515 items and 182 variables.

The initial runs on this data set revealed voting patterns along party lines. Among those that are sure they voted in the presidential election and said they think the Republicans mostly try to do what is best for the country and the Democrats mostly try to do what is best for their party, 91% voted for Trump, representing more than three-quarters of all Trump voters in the survey. For Hillary Clinton, the results were similar (with opposite party attributions).

However, the survey is composed of several parts and not all participants were asked the same set of questions to reduce the overall time per session. In a second analysis step, a subset marked as *nationally representative* with 4,913 registered voters was used. 1625 respondents (33.1%) said they voted for Donald Trump, and 2129 (43.3%) for Hillary Clinton.

The system trains a model that predicts the target value, so that it can visualize which and how input variables and ranges (*features*) correlate with the target,



Target: Q2016VOTE=(1) Donald Trump

Node 8 | Weight 0.991 | A 34.5% | C 34.6%

1,172 data rows above threshold, 111 below

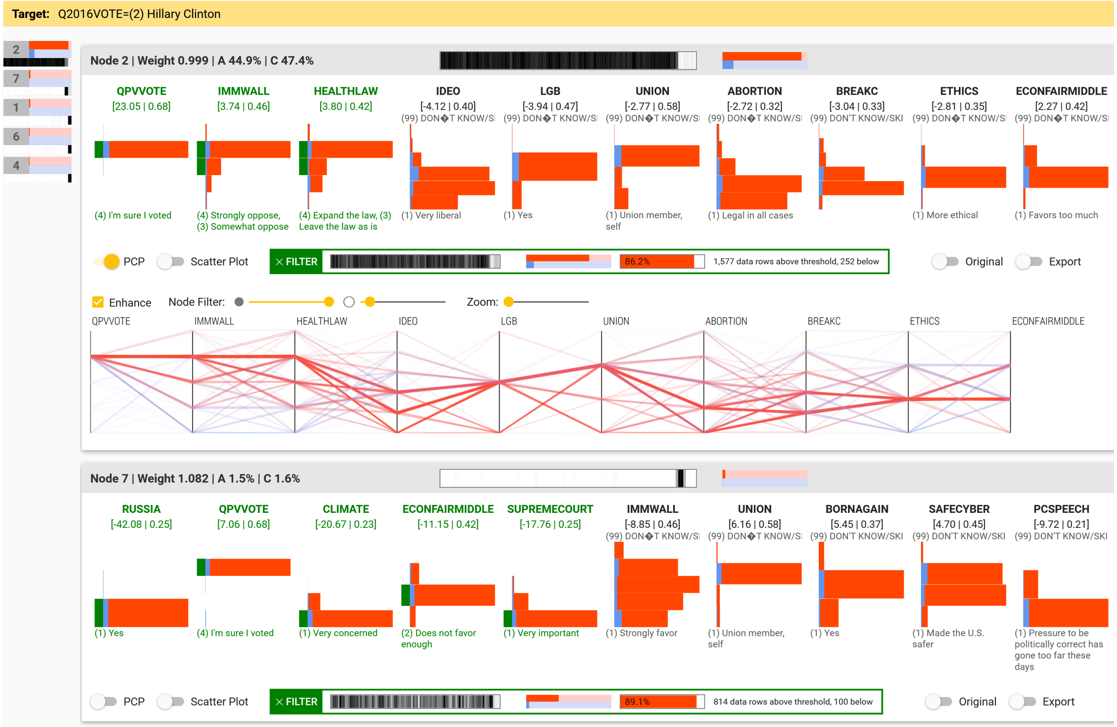
Node 1 | Weight 1.104 | A 1.0% | C 1.0%

258 data rows above threshold, 49 below

Figure 7.15 — Decomposition targeting Donald Trump voters. The data set is the AP VoteCast 2018 representative national survey comprising 4,913 respondents with 67 variables. The target variable is whether participants said they voted for Donald Trump in 2016 (1 yes, 0 no). Here, the analyst has selected several bars (in green) to define range filters.

but this also means that redundant features might not be used. The initial run on the subset showed that the participant's attitudes toward the parties are already a strong predictor for the voting behavior. Thus, more subtle (but redundant) correlation patterns might be missing when using all variables as input. To gain insights into voting behaviors based on demographics and policy attitudes, all variables relating to specific parties or persons were disabled (e.g., whether the respondent likes politician X). This reduced the number of variables to 67.

Figure 7.15 presents the result of VND for analyzing the (self-described) motives of Donald Trump voters, and Figure 7.16 for Hillary Clinton voters. The number of hidden nodes was set to 14 for each analysis. In contrast to the previously introduced cars and chip measurement data sets, the target variable is binary in this case. This means that each data item corresponding to a voter of the target candidate (Trump in Figure 7.15 and Clinton in Figure 7.16) counts as



Target: Q2016VOTE=(2) Hillary Clinton

Node 2 | Weight 0.999 | A 44.9% | C 47.4%

86.2% 1,577 data rows above threshold, 252 below

Node 7 | Weight 1.082 | A 1.5% | C 1.6%

89.1% 814 data rows above threshold, 100 below

Figure 7.16 — Decomposition targeting Hillary Clinton voters. The data set is the AP VoteCast 2018 representative national survey comprising 4,913 respondents with 67 variables. The target variable is whether participants said they voted for Hillary Clinton in 2016 (1 yes, 0 no). Here, the analyst has selected several bars (in green) to define range filters.

a high-value case (red bars in the stacked histograms and red lines the PCP). Conversely, if the participants did not vote for the target candidate (or did not vote at all), the respective items counts as low-value case (blue bars and lines).

The respective top nodes in both Figures show that attitudes towards the border wall and the Affordable Care Act (Obamacare) are strong predictors for the voting behavior of the participants. Among those that are sure they voted in the presidential election (QPVVOTE), favor the border wall (IMMWALL), think that the Affordable Care Act should be at least partially repealed (HEALTHLAW), are not gay or bisexual (LGB), and do not think that the Trump campaign coordinated with Russia during the election (RUSSIA), 91.3% said they voted for Trump, which represents 72.1% of all Trump voters in the survey. The parallel coordinate plot of Node 8 depicts the most salient attitudes of 'typical' Trump voters in the data set, for instance, positive attitudes toward the border wall and negative toward the Care Act. In the figure, the remaining data items were slightly faded in using the second node filter slider to better relate this to all respondents.

On the contrary, participants that oppose the border wall, think that the Care Act should be at least kept as it is, and are sure they voted in the election, 86% voted for Clinton as the range filter of Node 2 reveals. Furthermore, 89% of the respondents who think that the Trump campaign coordinated with Russia during the election (RUSSIA), are very concerned about climate change (CLIMATE), are sure they voted, think that the country's economic system does not favor the middle class enough (ECONFAIRMIDDLE), and think that the debate over Brett Kavanaugh's confirmation to the supreme court was very important (SUPREMECOURT) voted for Clinton, which represents about 39% of all Clinton voters in the data set.

Node 1 of the results investigating the motives of Donald Trump voters is interesting because it represents a small cluster that only partially overlaps with Node 8. Guided by the output of the node, the range filter was defined as follows (marked in green in Figure 7.15):

- **TRADENATIONALE:** 'Help'
  [whether the trade policies of Trump's administration would help the national economy]
- **QPVVOTE:** 'I'm sure I voted'
  [whether the respondent voted in the 2016 presidential election]
- **RACEREL:** 'Blacks have more advantages than whites' OR 'neither has much advantage over the other'
- **CLIMATE:** 'Somewhat concerned' OR 'very concerned'
- **SEX:** 'Women'

307 respondents match this filter, of which 258 (84%) said they voted for Trump, which is significantly higher than the 29.7% of Trump voters in the rest of the data set (Fisher's exact test, $p \ll 0.001$), This is a very interesting finding because when looking at some of the variables individually, one can observe totally different trends: 29.8% of the female respondents, and 20.4% of those at least somewhat concerned regarding climate change said they voted for Trump, which is in both cases less than the data set-wide average.

The final evaluation task with this data set was to investigate whether analysts would have found this insight with related approaches that first reduce the number of dimensions, e.g., using t-SNE. On the one hand, the use of DR

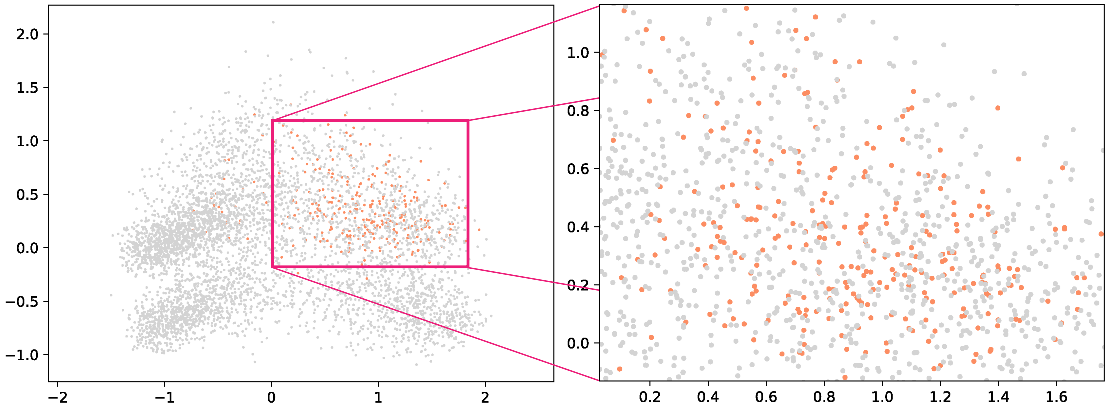


Figure 7.17 — PCA projection of the AP VoteCast 2018 representative national survey data set. The cluster of items matching the range filter (orange dots) does not appear as a well-separated visual cluster in the projection.

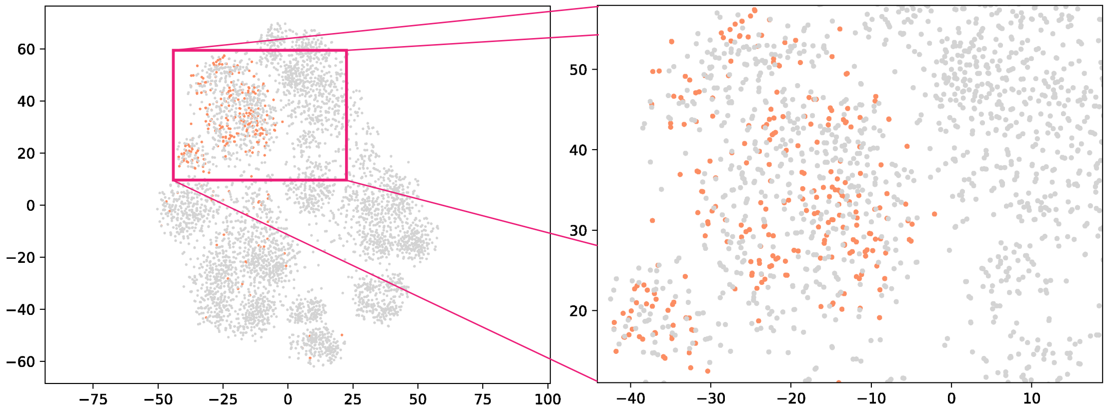


Figure 7.18 — t-SNE projection of the AP VoteCast 2018 representative national survey data set. The cluster of items matching the range filter (orange dots) does not appear as a well-separated visual cluster in the projection.

methods offers several advantages. Analysts can easily recognize visual clusters in the projection, and with fewer dimensions, subsequent processing steps are faster, which motivates the use of more complex algorithms that would not work for big, high-dimensional data sets. VND, on the other hand, clusters the data set in the high-dimensional space and incorporates the target value into the clustering process.

PCA, t-SNE, and UMAP (see Section 2.1.3) were applied to the data set of the 4,913 voters, with the same set of input variables and the same way of normalization. Figures 7.17,7.18,7.19 show the results. Dots corresponding to items that match the previously defined range filter are colored in orange.

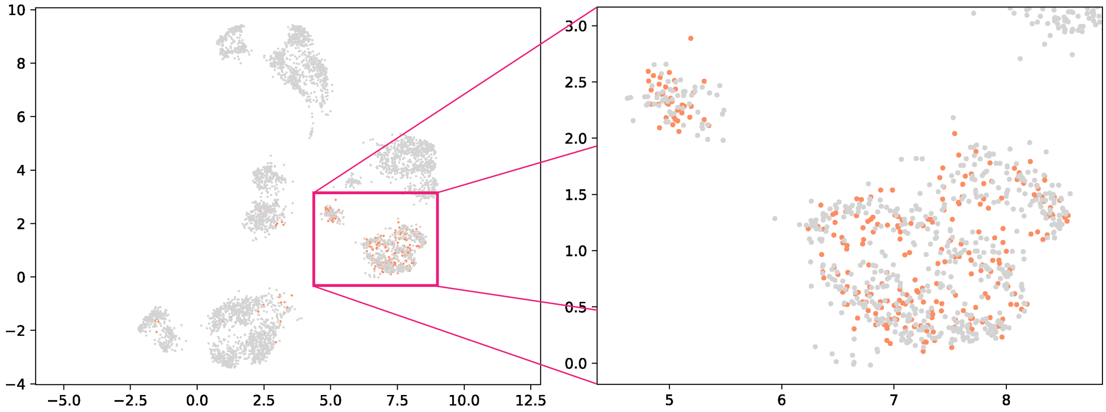


Figure 7.19 — UMAP projection of the AP VoteCast 2018 representative national survey data set. The cluster of items matching the range filter (orange dots) does not appear as a well-separated visual cluster in the projection.

PCA expectedly delivers the 'worst' result, in that the orange dots are scattered throughout the projection space. At first glance, the results from t-SNE and UMAP seem to have smaller regions with orange dots, but a detailed look reveals that between the orange dots there are a lot of gray ones. This behavior is consistent across several runs.

Hence, with the projections, it would be difficult for analysts to detect and explore a cluster comparable to the one that was found with VND. This shows that the proposed method can lead to interesting, significant insights that would be difficult to retrieve from methods relying on dimensionality reduction methods. However, this does not necessarily mean that the use of VND always leads to more insights than related DR methods, it just shows that it *can* detect more complex conditions compared to typical linear and nonlinear projection techniques.

Furthermore, the results show that a particular combination of several variables can exhibit much stronger (and even opposite) correlations with the target variable compared to individual variables or pairs, while at the same time still covering a large proportion of interesting high-value cases.

7.5.4 Qualitative Feedback

An earlier version of the system was presented to a domain expert and research fellow of the world's largest provider of automatic test equipment for preliminary feedback. In the session, the presenter introduced the system and showed the results of the data set the company provided (Section 7.5.3). The expert was impressed that the approach was capable to detect high-value conditions and that it supports the exploration of highly interesting combinations of variables. He was confident that this would greatly support the engineers in finding problematic conditions. However, he noted that there is a learning curve as it takes some time to understand the stacked histogram visualizations.

The expert made several suggestions on how to improve the utility of the approach. It should be possible to filter the data set by value ranges of certain variables (not just by nodes), the target threshold should be customizable, and it should be easily possible to remove or ignore data items based on already derived correlations to focus on the remaining, less-understood parts of the data set. Based on the feedback, the range filters were implemented and the target threshold was made adjustable. In addition, the node coverage histograms were developed.

7.6 Discussion

Visual Neural Decomposition can extract high-value conditions in multivariate data sets that contain many variables and it guides analysts to explore nonlinear multivariable relationships. The presented use cases show that the approach can also reveal significant relationships that are difficult to find with related methods, for instance, approaches applying dimensionality reduction techniques or linear models. With this novel approach, analysts do not need to iteratively select one or two variables at a time to start the process. Instead, clusters of similar behaviors are visualized separately based on the original dimensions that often have a semantic meaning in multivariate data sets. Additionally, the automatic analysis does not only cluster the results, but it also provides a ranking of the most important variables of each cluster. This helps users to grasp the meaning of each cluster. The provided interaction possibilities such as the range filter or node-specific parallel coordinate plots enable analysts to validate their hypotheses and build trust in what they see.

The proposed technique also has certain limitations. While it scales to hundreds of variables, it was not designed for the analysis of very high-dimensional data sets, for instance, images with millions of pixels. However, in this case, it is often not helpful to explain certain conditions based on single pixel values. A shortcoming is that analysts have to define hyper-parameters, for instance, the number of nodes, and each run can lead to a slightly different output. Nevertheless, the performed analysis (Section 7.5.2) shows that the method is not very sensitive to the specified parameters. Generally, more nodes and longer training are better, but also computationally more expensive. In the presented use cases, the training only took seconds, though.

The regularization constrains the solution space, but there can still be several equally valid definitions of clusters, particularly if the variables are not independent of each other. The goal of VND is to visually explain high-value cases, hence, if one input is already enough to perfectly predict the target then VND probably will not extract additional (somewhat redundant) cases.

Finally, it is important to realize that the approach visualizes interesting conditions that appear *in the data*. If analysts want to generalize such findings, for instance, whether an observed voting behavior applies to the general population, they still need to perform statistical tests for significance.

7.7 Summary

Approaches for the visual analysis of multivariate data sets with well over ten dimensions have to tackle multiple challenges. The human visual perception is limited to three dimensions and it is computationally demanding to determine multidimensional correlations on large data sets. This chapter therefore proposed a novel method to scale this analysis by visually explaining multivariate data sets using neural networks. The idea is to decompose the data set into components as represented by the hidden nodes and provide node-specific views of the data set. The training process supports large data sets and the card-based visual interface helps analysts better digest various different relationships in the data set, even if the total number of variables is high. An integrated approach was developed to configure the data, train the model, visualize the results, and let analysts validate hypotheses with interaction mechanisms.

Chapter 8

Conclusion and Outlook

The main aim of this work was to investigate and develop scalable approaches for the visual analysis of textual and multivariate data sets. As noted in the beginning of this thesis, *scaling* is a complex concept. Not only does it entail making an approach *executable* on larger data sets (either by increasing the computational capacity or the algorithmic efficiency), but it also entails dealing with the additional challenges of aggregating lots of data in a meaningful way and enabling analysts to interactively explore more complex patterns and relationships. Only then can the analysis of large or high-dimensional data sets unfold its full potential. In general, this is a very complex, multi-layered, and demanding issue that cannot easily be solved comprehensively. Hence, this thesis contributes in different ways to making large-scale analyses feasible for certain analytical goals and use cases.

8.1 Contributions and Limitations

Chapter 3 introduced two new methods for visually aggregating large text collections while still providing sufficient context so that analysts can make sense of the content. The first method, ELSKE, efficiently extracts keywords and (potentially longer) keyphrases to provide brief visual summaries of individual documents but also large collections. The second one extracts shortened sequences of phrases (quotes) in a hierarchical manner to provide more context-rich yet compact summaries of short texts such as tweets. The resulting quotes are typically more verbose compared to keyphrases so that analysts can better quantify the extent of more concrete statements, given that they are similarly phrased. Both methods exploit the size of the data set to extract reasonably meaningful phrases and sequences without having to rely on expensive grammatical analyses. In addition, a key design goal was that they are efficient enough to continuously analyze streaming data in real-time. However, the frequency- and heuristics-based approach may sometimes lead to incomplete or misleading aggregations, which is a common challenge with content summarization approaches.

PyramidTags is a context-aware, word order-aware, and date-aware tag map that was presented in Chapter 4 for exploring large time-stamped document collections such as news reports. First, relevant tags are extracted with ELSKE and their co-occurrence behavior is analyzed. Then, the tags are placed onto a map using a triangular layout such that analysts can understand the temporal evolution of tags and visual clusters. Additionally, co-occurring tags should be placed nearby and a dominant word order should be respected. In contrast to previous work, this visual encoding enables analysts to explore themes and how they evolve over time in large collections without resorting to animations or pre-computed topics. In combination with the provided interaction possibilities, the approach is able to visually aggregate a large proportion of hundreds of thousands of documents with just a few hundred tags. The approach works best when the underlying documents have different occurrence patterns (e.g., reports of events that happened on a particular day), which is often the case for news articles and social media posts. If the temporal development across the data items is similar or random, most tags will only be placed in a few regions, rendering the triangular layout useless. Another limitation is that the reduction to a two-dimensional representation can also lead to false friends, that is, neighboring tags that do not relate to each other. Thus, analysts have to hover over the tags to confirm or reject their hypotheses.

Clustering divides documents into groups, which helps to provide analysts a thematic overview of the content. For the interactive analysis of large collections or streaming posts, however, we need an efficient and explainable clustering algorithm that also supports incremental updates. Chapter 5 introduces several strategies to accelerate the Spherical *k*-Means algorithm on sparse document representations. It further proposes the Dynamic Spherical *k*-Means algorithm that supports incremental updates and dynamically chooses an appropriate number of clusters.

The proposed explainable dynamic clustering algorithm powers a visual analytics approach for the real-time monitoring of social media posts, which is introduced in Chapter 6. In combination with the ELSKE algorithm, it equips analysts with interactive visualizations that enable them to gain a continuous overview of currently published tweets, monitor specific topics of interest at varying levels of detail with digestible aggregations, and dive deeper into topics by increasing the resolution and specificity of the analysis. Compared to earlier work, the approach does not rely on additional meta-data such as the geolocation of posts and avoids computationally expensive and delay-inducing event detection mechanisms. However, one of the main limitations of the approach is that it mainly operates on the textual content of the posts. The clustering process currently does not consider media content (images or videos) for efficiency reasons.

It is difficult to extract and visualize multivariate relationships in data sets due to the curse of dimensionality and the limited human perception. Chapter 7 introduces Visual Neural Decomposition to tackle this problem. Using a modified neural network architecture, it aims to disentangle the problem of how the independent variables influence a chosen target variable into separate clusters that can then be visualized independently of one another. The stacked histograms visualization, range filters, and filtered parallel coordinates as well as scatter plots further help analysts to recognize and understand the combinations that lead to high values of the target value. One of the main advantages of the approach is that it can extract more complex relationships, but it also scales to larger data sets and many dimensions. However, with some very demanding data sets, the resulting analysis can occasionally lead to visualizations that are more difficult to interpret immediately.

The contributions of this work can be generalized into the concept of exploiting artificial intelligence for visually explaining data sets. Section 2.5 formalizes this concept with the AIX process. We train a model on or fit a function to the input data not for the sake of predicting values, but rather to understand relationships of the underlying data by visualizing the inner workings of the model or function. That is, we are not primarily interested in the outputs of the model, but in how the model captured what it has learned. This can be applied to supervised as well as unsupervised methods. For instance, the Visual Neural Decomposition approach is an example of a supervised training procedure to extract and visualize multivariate relationships, whereas PyramidTags and the dynamic clustering process are based on unsupervised machine learning techniques. The AIX process is intended to be a model for characterizing existing techniques as well as for inciting and steering novel visual analytics approaches that scale to large data sets.

8.2 Open Challenges

Research is like the Sigmoid function, you can always progress but you will never reach the ceiling. This thesis made several contributions to enable large-scale analyses of textual and multivariate data sets combining machine learning and visualization, but there are still open challenges for future work.

In recent years, there has been tremendous progress in deep learning-based techniques for natural language and image processing. With these techniques, visual analytics approaches may provide more concise and more accurate summaries of documents, for instance, but the immense computational cost of invoking these models on non-high-performance computing hardware has yet to be addressed. In addition, deep learning-based models typically focus on sequences with up to thousands of tokens, which makes them less suited for visually aggregating entire collections. One promising strategy for reducing the computational burden is to research sparser architectures that can achieve similar performance but with fewer parameters.

Another challenge is that documents are becoming more and more media-rich. For instance, a significant proportion of tweets only contain one or a few words alongside an image or video, which makes it increasingly difficult to process and analyze vast amounts of posts interactively. A similar problem arises when posts reference and link external content. Thus, it is becoming increasingly important to develop visual analytics systems that do not solely rely on pre-processed data sources, but are also equipped with analysis methods that continuously enrich the collected data in real-time.

In the case of analyzing tabular data, it is still difficult to visualize truly multivariate relationships without allowing ambiguous interpretations, particularly if these relationships are composed of irregular or non-smooth patterns. Furthermore, if there are several intertwined effects that cause a correlation in the data, it becomes increasingly challenging to disentangle these effects into separate cases for an easier visual interpretation. More research is needed whether and how regularization techniques can reduce ambiguity and help to extract overlapping effects.

Visual Neural Decomposition aims to explain multivariate data sets with one dependent variable. While separate analysis steps can be performed on several such dependent variables, analysts may also want to examine relationships that involve not only multiple independent variables but also multiple dependent variables. One could extend the neural network architecture to predict several target variables, but it remains challenging how to adapt the visualizations such that they can express non-linear N-to-M relationships.

The analysis of large and complex data sets promises important new findings that can have a real and lasting impact on future generations. Despite continuous advances in pure computing power, however, developing both efficient and powerful methods for the visual analysis of textual and multivariate data sets is still a challenging endeavor that has many obstacles to overcome. This thesis tackled some of these challenges in order to make large-scale analyses feasible in many use cases.

Bibliography

F. Abel, C. Hauff, G. J. Houben, K. Tao, and R. Stronkman. Twitcident: Fighting fire with information from Social Web streams. In *Proceedings of the 21st Annual Conference on World Wide Web Companion, WWW 2012*, pages 305–308, 2012. 31

M. R. Ackermann, C. Lammersen, M. Märtens, C. Raupach, C. Sohler, and K. Swierkot. StreamKM++: A clustering algorithm for data streams. In *Proceedings of the 12th Workshop on Algorithm Engineering and Experiments, ALENEX 2010*, pages 173–187, 2010. 16

I. Adä, K. Thiel, and M. R. Berthold. Distance aware tag clouds. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, pages 2316–2322, 2010. 26

C. C. Aggarwal and C. X. Zhai. A survey of text clustering algorithms. In *Mining Text Data*, pages 77–128. Springer, Boston, MA, 2012. 12, 13

N. Ailon, R. Jaiswal, and C. Monteleoni. Streaming k-means approximation. In *Proceedings of the Advances in Neural Information Processing Systems 22*, 2009. 16

Z. Alami Merrouni, B. Frikh, and B. Ouhbi. Automatic keyphrase extraction: a survey and trends. *Journal of Intelligent Information Systems*, 54(2):391–424, 2020. 48

S. Alemzadeh, T. Hielscher, U. Niemann, L. Cibulski, T. Ittermann, H. Völzke, M. Spiliopoulou, and B. Preim. Subpopulation Discovery and Validation in Epidemiological Data. In M. Sedlmair and C. Tominski, editors, *Proceedings of the EuroVis Workshop on Visual Analytics, EuroVA 2017*. The Eurographics Association, 2017. 37

A. B. Alencar, M. C. F. de Oliveira, and F. V. Paulovich. Seeing beyond reading: a survey on visual text analytics. *WIREs Data Mining and Knowledge Discovery*, 2(6):476–492, 2012. 24

E. Alexander and M. Gleicher. Assessing topic representations for GIST-forming. In *Proceedings of the Workshop on Advanced Visual Interfaces AVI*, pages 100–107, 2016. 29

E. Alexander, J. Kohlmann, R. Valenza, M. Witmore, and M. Gleicher. Serendip: Topic model-driven visual exploration of text corpora. In *Proceedings of the 2014 IEEE Conference on Visual Analytics Science and Technology, VAST 2014*, pages 173–182, 2015. 28

M. Allahyari, S. Pouriyeh, M. Assefi, S. Safaei, E. D. Trippe, J. B. Gutierrez, and K. Kochut. A Brief Survey of Text Mining: Classification, Clustering and Extraction Techniques, 2017. [Online]. Available: `http://arxiv.org/abs/1707.02919`. 12, 13

R. B. Allen and R. Sieczkiewicz. How historians use historical newspapers. In *Proceedings of the ASIST Annual Meeting*, volume 47, pages 1–4. John Wiley and Sons, nov 2010. 23

J. Alsakran, Y. Chen, Y. Zhao, J. Yang, and D. Luo. STREAMIT: Dynamic visualization and interactive exploration of text streams. In *Proceedings of the 2011 IEEE Pacific Visualization Symposium, PacificVis 2011*, pages 131–138, 2011. 24, 25, 31, 32

R. Amar and J. Stasko. A knowledge task-based framework for design and evaluation of information visualizations. In *Proceedings of the 2004 IEEE Symposium on Information Visualization, InfoVis 2004*, pages 143–149, 2004. 34

D. Archambault, D. Greene, P. Cunningham, and N. Hurley. ThemeCrowds: Multiresolution summaries of twitter usage. In *Proceedings of the International Conference on Information and Knowledge Management*, pages 77–84, 2011. 31

A. O. Artero, M. C. Ferreira De Oliveira, and H. Levkowitz. Uncovering clusters in crowded parallel coordinates visualizations. In *Proceedings of the 2004 IEEE Symposium on Information Visualization, InfoVis 2004*, pages 81–88, 2004. 35

D. Arthur and S. Vassilvitskii. K-means++: The advantages of careful seeding. Technical Report 2006-13, Stanford InfoLab, 2007. 15

I. Assent, R. Krieger, E. Müller, and T. Seidl. Visa. *ACM SIGKDD Explorations Newsletter*, 9(2):5–12, 2007. 37

S. Barlowe, T. Zhang, Y. Liu, J. Yang, and D. Jacobs. Multivariate visual explanation for high dimensional datasets. In *Proceedings of the 2008 IEEE Symposium on Visual Analytics Science and Technology, VAST 2008*, pages 147–154, 2008. 35, 38

A. Barredo Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. Garcia, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila, and F. Herrera. Explainable Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 58:82–115, 2020. 44

S. Bateman, C. Gutwin, and M. Nacenta. Seeing things in the clouds: the effect of visual features on tag cloud selections. In *Proceedings of the nineteenth ACM conference on Hypertext and hypermedia*, pages 193–202, 2008. 25

D. Bau, J.-Y. Zhu, H. Strobelt, A. Lapedriza, B. Zhou, and A. Torralba. Understanding the role of individual units in a deep neural network. *Proceedings of the National Academy of Sciences*, 117(48):30071–30078, 2020. 41

M. Behrisch, F. Korkmaz, L. Shao, and T. Schreck. Feedback-driven interactive exploration of large multidimensional data supported by visual classifier. In *Proceedings of the 2014 IEEE Conference on Visual Analytics Science and Technology, VAST 2014*, pages 43–52, 2015. 37

G. Beigi, X. Hu, R. Maciejewski, and H. Liu. An overview of sentiment analysis in social media and its applications in disaster relief. In W. Pedrycz and S.-M. Chen, editors, *Sentiment Analysis and Ontology Engineering: An Environment of Computational Intelligence*, volume 639, pages 313–340. Springer, Cham, 2016. 23

J. L. Bentley. Multidimensional Binary Search Trees Used for Associative Searching. *Communications of the ACM*, 18(9):509–517, 1975. 102

J. Bernard, M. Steiger, S. Widmer, H. Lücke-Tieke, T. May, and J. Kohlhammer. Visual-interactive exploration of interesting multivariate relations in mixed research data sets. *Computer Graphics Forum*, 33(3):291–300, 2014. 39

C. Binucci, W. Didimo, and E. Spataro. Fully dynamic semantic word clouds. In *Proceedings of the 7th International Conference on Information, Intelligence, Systems and Applications, IISA 2016*, pages 1–6, 2016. 27

C. M. Bishop. Neural networks and their applications. *Review of scientific instruments*, 65(6):1803–1832, 1994. 19

D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003. 14

C. Blundell, Y. W. Teh, and K. A. Heller. Bayesian rose trees. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence, UAI 2010*, pages 65–72, 2010. 16

H. Bosch, D. Thom, F. Heimerl, E. Puttmann, S. Koch, R. Kruger, M. Worner, and T. Ertl. ScatterBlogs2: Real-time monitoring of microblog messages through user-guided filtering. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2022–2031, 2013. 32, 33

A. Bougouin and F. Boudin. TopicRank : Graph-Based Topic Ranking for Keyphrase Extraction. In *Proc. IJCNLP 2013*, number October, pages 543–551, 2013. 48

V. Braverman, A. Meyerson, R. Ostrovsky, A. Roytman, M. Shindler, and B. Tagiku. Streaming k-means on well-clusterable data. In *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms*, 2011. 16

M. Burch, S. Lohmann, D. Pompe, and D. Weiskopf. Prefix tag clouds. In *Proceedings of the International Conference on Information Visualisation*, pages 45–50, 2013. 27

R. Campos, V. Mangaravite, A. Pasquali, A. Jorge, C. Nunes, and A. Jatowt. YAKE! Keyword extraction from single documents using multiple local features. *Information Sciences*, 509:257–289, 2020. 48

N. Cao, D. Gotz, J. Sun, and H. Qu. DICON: Interactive visual analysis of multidimensional clusters. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2581–2590, 2011. 37

N. Cao, Y. R. Lin, X. Sun, D. Lazer, S. Liu, and H. Qu. Whisper: Tracing the spatiotemporal process of information diffusion in real time. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2649–2658, 2012. 33

S. K. Card, J. D. Mackinlay, and B. Shneiderman. Information visualization: using vision to think. In *Readings in information visualization: using vision to think*. Morgan Kaufmann, 1999. 22

C. Carpineto, S. Osiski, G. Romano, and D. Weiss. A survey of web clustering engines. *ACM Computing Surveys*, 41(3), 2009. 29

D. B. Carr, R. J. Littlefield, W. L. Nicholson, and J. S. Littlefield. Scatterplot Matrix Techniques for Large N. *Journal of the American Statistical Association*, 82(398):424, 1987. 35

J. Chae, D. Thom, H. Bosch, Y. Jang, R. Maciejewski, D. S. Ebert, and T. Ertl. Spatiotemporal social media analytics for abnormal event detection and examination using seasonal-trend decomposition. In *Proceedings of the 2012 IEEE Conference on Visual Analytics Science and Technology, VAST 2012*, pages 143–152, 2012. 33

D. Chakrabarti, R. Kumar, and A. Tomkins. Evolutionary clustering. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, volume 2006, pages 554–560, 2006. 16

A. Chatzimparmpas, R. M. Martins, I. Jusufi, and A. Kerren. A survey of surveys on the use of visualization for interpreting machine learning models. *Information Visualization*, 19(3):207–233, 2020a. 44

A. Chatzimparmpas, R. M. Martins, I. Jusufi, K. Kucher, F. Rossi, and A. Kerren. The State of the Art in Enhancing Trust in Machine Learning Models with the Use of Visualizations. *Computer Graphics Forum*, 39(3):713–756, 2020b. 44

A. Chatzimparmpas, R. M. Martins, and A. Kerren. T-viSNE: Interactive Assessment and Interpretation of t-SNE Projections. *IEEE Transactions on Visualization and Computer Graphics*, 26(8):2696–2714, 2020c. 36

J. Chen, X. Zhang, Y. Wu, Z. Yan, and Z. Li. Keyphrase generation with correlation constraints. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018*, pages 4057–4066, 2020. 55

S. Chen, L. Lin, and X. Yuan. Social Media Visual Analytics. *Computer Graphics Forum*, 36(3):563–587, 2017. 30

S. Chen, N. Andrienko, G. Andrienko, J. Li, and X. Yuan. Co-Bridges: Pairwise Visual Connection and Comparison for Multi-item Data Streams. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1612–1622, 2021. 31

Y. Chen, L. Wang, M. Dong, and J. Hua. Exemplar-based Visualization of Large Document Corpus. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1161–1168, 2009. 24

S. Cheng, W. Xu, and K. Mueller. RadViz Deluxe: An Attribute-Aware Display for Multivariate Data. *Processes*, 5(4):75, 2017. 37

M. T. Chi, S. S. Lin, S. Y. Chen, C. H. Lin, and T. Y. Lee. Morphable Word Clouds for Time-Varying Text Data Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 21(12):1415–1426, 2015. 26, 68

J. Choo and S. Liu. Visual Analytics for Explainable Deep Learning. *IEEE Computer Graphics and Applications*, 38(4):84–92, 2018. 40, 44

J. Choo, C. Lee, C. K. Reddy, and H. Park. UTOPIAN: User-driven topic modeling based on interactive nonnegative matrix factorization. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):1992–2001, 2013. 14, 29

B. Chopard and M. Tomassini. Particle swarm optimization. *Natural Computing Series*, 4:97–102, 2018. 83

J. Chuang, C. D. Manning, and J. Heer. “Without the clutter of unimportant words”. *ACM Transactions on Computer-Human Interaction*, 19(3):1–29, 2012a. 25, 70

J. Chuang, C. D. Manning, and J. Heer. Termite: Visualization techniques for assessing textual topic models. In *Proceedings of the Workshop on Advanced Visual Interfaces AVI*, pages 74–77, 2012b. 28, 68, 70

J. Chuang, D. Ramage, C. D. Manning, and J. Heer. Interpretation and trust: Designing model-driven visualizations for text analysis. In *Proceedings of the Conference on Human Factors in Computing Systems*, pages 443–452, 2012c. 29

W. Chung. BizPro: Extracting and categorizing business intelligence factors from textual news articles. *International Journal of Information Management*, 34 (2):272–284, 2014. 23

C. Collins, F. B. Viégas, and M. Wattenberg. Parallel tag clouds to explore and analyze faceted text corpora. In *Proceedings of the 2009 IEEE Symposium on Visual Analytics Science and Technology, VAST 2009*, pages 91–98, 2009. 26

M. A. A. Cox and T. F. Cox. *Multidimensional Scaling*, pages 315–347. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. 16, 35

W. Cui, Y. Wu, S. Liu, F. Wei, M. Zhou, and H. Qu. Context-preserving, dynamic word cloud visualization. *IEEE Computer Graphics and Applications*, 30(6): 42–53, 2010. 26

W. Cui, S. Liu, L. Tan, C. Shi, Y. Song, Z. Gao, H. Qu, and X. Tong. TextFlow: Towards Better Understanding of Evolving Topics in Text. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2412–2421, 2011. 30

W. Cui, S. Liu, Z. Wu, and H. Wei. How hierarchical topics evolve in large text corpora. *IEEE Transactions on Visualization and Computer Graphics*, 20(12): 2281–2290, 2014. 30

C. Culy and V. Lyding. Double tree: An advanced KWIC visualization for expert users. In *Proceedings of the International Conference on Information Visualisation*, pages 98–103, 2010. 57

R. R. O. da Silva, P. E. Rauber, R. M. Martins, R. Minghim, and A. C. Telea. Attribute-based Visual Explanation of Multidimensional Projections. In E. Bertini and J. C. Roberts, editors, *Proceedings of the EuroVis Workshop on Visual Analytics, EuroVA 2015*. The Eurographics Association, 2015. 36

D. L. Davies and D. W. Bouldin. A Cluster Separation Measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1(2):224–227, 1979. 111

J. Devlin, M. W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL HLT 2019*, volume 1, pages 4171–4186, 2019. 12

I. S. Dhillon and D. S. Modha. Concept decompositions for large sparse text data using clustering. *Machine Learning*, 42(1-2):143–175, 2001. 14, 15, 101

N. Diakopoulos, M. Naaman, and F. Kivran-Swaine. Diamonds in the rough: Social media visual analytics for journalistic inquiry. In *Proceedings of the 2010 IEEE Symposium on Visual Analytics Science and Technology, VAST 2010*, pages 115–122, 2010. 30

D. Dingen, M. Van't Veer, P. Houthuizen, E. H. Mestrom, E. H. Korsten, A. R. Bouwman, and J. Van Wijk. RegressionExplorer: Interactive Exploration of Logistic Regression Models with Subgroup Analysis. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):246–255, 2019. 37, 39

G. Doquire and M. Verleysen. A comparison of multivariate mutual information estimators for feature selection. In *Proceedings of the 1st International Conference on Pattern Recognition Applications and Methods, ICPRAM 2012*, volume 1, pages 176–185, 2012. 34

M. Dörk, S. Carpendale, C. Collins, and C. Williamson. VisGets: Coordinated visualizations for web-based information exploration and discovery. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1205–1212, 2008. 26, 30

M. Dörk, D. Gruen, C. Williamson, and S. Carpendale. A Visual backchannel for large-scale events. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1129–1138, 2010. 31

W. Dou, X. Wang, R. Chang, and W. Ribarsky. ParallelTopics: A probabilistic approach to exploring document collections. In *Proceedings of the 2011 IEEE Symposium on Visual Analytics Science and Technology, VAST 2011*, pages 231–240, 2011. 28, 30

W. Dou, X. Wang, D. Skau, W. Ribarsky, and M. X. Zhou. LeadLine: Interactive visual analysis of text data through event identification and exploration. In *Proceedings of the 2012 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 93–102, 2012. 31

W. Dou, L. Yu, X. Wang, Z. Ma, and W. Ribarsky. HierarchicalTopics: Visually exploring large text collections using topic hierarchies. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2002–2011, 2013. 28

M. Dredze, H. M. Wallach, D. Puller, and F. Pereira. Generating summary keywords for emails using topics. In *Proceedings of the 2008 International Conference on Intelligent User Interfaces, IUI 2008*, 2008. 28

G. Dreyfus. *Neural networks: Methodology and applications*. Springer Science & Business Media, 2005. 17, 19

C. Eichner, H. Schumann, and C. Tominski. Making Parameter Dependencies of Time-Series Segmentation Visually Understandable. *Computer Graphics Forum*, 39(1), 2020. 38

M. El-Assady, V. Gold, C. Acevedo, C. Collins, and D. Keim. ConToVi: Multi-Party Conversation Exploration using Topic-Space Views. *Computer Graphics Forum*, 35(3):431–440, 2016. 29

S. R. El-Beltagy and A. Rafea. KP-Miner: A keyphrase extraction system for English and Arabic documents. *Information Systems*, 34(1):132–144, 2009. 48

C. Elkan. Using the Triangle Inequality to Accelerate k-Means. In *Proceedings, Twentieth International Conference on Machine Learning*, volume 1, pages 147–153, 2003. 102

A. Endert, R. Burtner, N. Cramer, R. Perko, S. Hampton, and K. Cook. Typograph: Multiscale spatial exploration of text documents. In *Proceedings of the 2013 IEEE International Conference on Big Data, Big Data 2013*, pages 17–24, 2013. 26

Y. Endo and S. Miyamoto. Spherical k-means++ clustering. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 9321, pages 103–114, 2015. 110

U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery in databases. *AI Magazine*, 17(3):37–53, 1996. 19, 20, 22, 43

C. Felix, S. Franconeri, and E. Bertini. Taking Word Clouds Apart: An Empirical Investigation of the Design Space for Keyword Summaries. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):657–666, 2018. 25

J. Ferwerda, A. van Saase, B. Unger, and M. Getzner. Estimating money laundering flows with a gravity model-based simulation. *Scientific Reports*, 10 (1):18552, 2020. 2

T. Fujiwara, O. H. Kwon, and K. L. Ma. Supporting Analysis of Dimensionality Reduction Results with Contrastive Learning. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):45–55, 2020. 36

S. Gad, W. Javed, S. Ghani, N. Elmqvist, T. Ewing, K. N. Hampton, and N. Ramakrishnan. ThemeDelta: Dynamic segmentations over temporal topic models. *IEEE Transactions on Visualization and Computer Graphics*, 21(5):672–685, 2015. 30

M. Gambhir and V. Gupta. Recent automatic text summarization techniques: a survey. *Artificial Intelligence Review*, 47(1):1–66, 2017. 47

E. R. Gansner, Y. Hu, and S. North. Interactive visualization of streaming text data with dynamic maps. *Journal of Graph Algorithms and Applications*, 17(4): 515–540, 2013. 24, 25, 31

Z. J. Gao, Y. Song, S. Liu, H. Wang, H. Wei, Y. Chen, and W. Cui. Tracking and connecting topics via incremental hierarchical Dirichlet processes. In *Proceedings of the 2011 IEEE International Conference on Data Mining, ICDM 2011*, pages 1056–1061, 2011. 15

L. Garrison, J. Müller, S. Schreiber, S. Oeltze-Jafra, H. Hauser, and S. Bruckner. DimLift: Interactive Hierarchical Data Exploration Through Dimensional Bundling. *IEEE Transactions on Visualization and Computer Graphics*, 27(6): 2908–2922, 2021. 36

J. E. Gentle, L. Kaufman, and P. J. Rousseuw. *Finding Groups in Data: An Introduction to Cluster Analysis.*, volume 47. John Wiley and Sons, 1991. 12

N. Gershon, S. G. Eick, and S. Card. Information Visualization. *Interactions*, 5 (2):9–15, 1998. 20

R. Gil-García and A. Pons-Porrata. Dynamic hierarchical algorithms for document clustering. *Pattern Recognition Letters*, 31(6):469–477, 2010. 15

M. Gleicher. Explainers: Expert explorations with crafted projections. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2042–2051, 2013. 36

M. Gleicher. A Framework for Considering Comprehensibility in Modeling. *Big data*, 4(2):75–88, jun 2016. 43, 44

L. B. Godfrey and M. S. Gashler. Neural decomposition of time-series data. In *Proceedings of the 2017 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2017*, volume 2017-Janua, pages 2796–2801, 2017. 136

S. Goodwin, J. Dykes, A. Slingsby, and C. Turkay. Visualizing Multiple Variables Across Scale and Geography. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):599–608, 2016. 37

S. Günnemann, H. Kremer, I. Färber, and T. Seidl. MCExplorer: Interactive exploration of multiple (subspace) clustering solutions. In *Proceedings of the 2010 IEEE International Conference on Data Mining, ICDM 2010*, pages 1387–1390, 2010. 37

M. J. Halvey and M. T. Keane. An assessment of tag presentation techniques. In *Proceedings of the 16th International Conference on World Wide Web*, pages 1313–1314, 2007. 68

K. S. Hasan and V. Ng. Conundrums in unsupervised keyphrase extraction: Making sense of the state-of-the-art. In *Proceedings of the 23rd International Conference on Computational Linguistics, Coling 2010*, volume 2, pages 365–373, 2010. 48, 49, 50, 55

S. Hassan, J. Sänger, and G. Pernul. SoDA: Dynamic visual analytics of big social data. In *Proceedings of the 2014 International Conference on Big Data and Smart Computing, BIGCOMP 2014*, pages 183–188, 2014. 30

T. Hastie and R. Tibshirani. Generalized additive models. *Statistical Science*, 1 (3):297–310, 1986. 43

T. Hastie and R. Tibshirani. Generalized Additive Models: Some Applications. *Journal of the American Statistical Association*, 82(398):371, 1987. 43

S. Havre, E. Hetzler, P. Whitney, and L. Nowell. ThemeRiver: Visualizing thematic changes in large document collections. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):9–20, 2002. 29, 30

M. A. Hearst, E. Pedersen, L. Patil, E. Lee, P. Laskowski, and S. Franconeri. An Evaluation of Semantically Grouped Word Cloud Designs. *IEEE Transactions on Visualization and Computer Graphics*, 26(9):2748–2761, 2020. 68, 69, 70

F. Heimerl, S. Koch, H. Bosch, and T. Ertl. Visual Classifier Training for Text Document Retrieval. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2839–2848, 2012. 43

F. Heimerl, S. Lohmann, S. Lange, and T. Ertl. Word cloud explorer: Text analytics based on word clouds. In *Proceedings of the Annual Hawaii International Conference on System Sciences*, pages 1833–1842, 2014. 26, 68

F. Heimerl, Q. Han, S. Koch, and T. Ertl. CiteRivers: Visual Analytics of Citation Patterns. *IEEE Transactions on Visualization and Computer Graphics*, 22 (1):190–199, 2016. 30

F. Heimerl, M. John, Q. Han, S. Koch, and T. Ertl. DocuCompass: Effective exploration of document landscapes. In *Proceedings of the 2016 IEEE Conference on Visual Analytics Science and Technology, VAST 2016*, pages 11–20, 2017. 24

J. Heinrich and D. Weiskopf. State of the Art of Parallel Coordinates. In *Proceedings of the Eurographics Conference on Visualization, EuroVis 2013*, pages 95–116, 2013. 35

J. Heinrich, Y. Luo, A. E. Kirkpatrick, H. Zhang, and D. Weiskopf. Evaluation of a bundling technique for parallel coordinates. *arXiv preprint arXiv:1109.6073*, 2011. 35

K. A. Heller and Z. Ghahramani. Bayesian hierarchical clustering. In *Proceedings of the 22nd International Conference on Machine Learning, ICML 2005*, pages 297–304, 2005. 16

J. M. Helm, A. M. Swiergosz, H. S. Haeberle, J. M. Karnuta, J. L. Schaffer, V. E. Krebs, A. I. Spitzer, and P. N. Ramkumar. Machine Learning and Artificial Intelligence: Definitions, Applications, and Future Directions. *Current reviews in musculoskeletal medicine*, 13(1):69–76, feb 2020. 10

S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 1997. 17

D. L. Hoffman and M. Fodor. Can you measure the ROI of your social media marketing? *MIT Sloan management review*, 52(1):41, 2010. 23

E. Hoque and G. Carenini. Interactive topic hierarchy revision for exploring a collection of online conversations. *Information Visualization*, 18(3):318–338, 2019. 29

T. Horsmann, N. Erbs, and T. Zesch. Fast or Accurate ? – A Comparative Evaluation of PoS Tagging Models. *Proceedings of the International Conference of the German Society for Computational Linguistics and Language Technology (GSCL-2015)*, 2015. 48, 54

M. Hu, S. Liu, F. Wei, Y. Wu, J. Stasko, and K. L. Ma. Breaking news on Twitter. In *Proceedings of the Conference on Human Factors in Computing Systems*, pages 2751–2754, 2012. 23

M. Hu, K. Wongsuphasawat, and J. Stasko. Visualizing Social Media Content with SentenTree. *IEEE Transactions on Visualization and Computer Graphics*, 23 (1):621–630, 2017. 28, 31, 57, 68, 70

A. Hulth. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing, EMNLP 2003*, pages 216–223, 2003. 55

M. Hund, D. Böhm, W. Sturm, M. Sedlmair, T. Schreck, T. Ullrich, D. A. Keim, L. Majnaric, and A. Holzinger. Visual analytics for concept exploration in subspaces of patient groups: Making sense of complex datasets with the Doctor-in-the-loop. *Brain Informatics*, 3(4):233–247, 2016. 37

M. Imran, C. Castillo, F. Diaz, and S. Vieweg. Processing social media messages in Mass Emergency: A survey. *ACM Computing Surveys*, 47(4):67:1—-67:38, 2015. 48

A. Inselberg. The plane with parallel coordinates. *The Visual Computer*, 1(4): 69–91, 1985. 35

D. Jackle, M. Hund, M. Behrisch, D. A. Keim, and T. Schreck. Pattern Trails: Visual Analysis of Pattern Transitions in Subspaces. In *Proceedings of the 2017 IEEE Conference on Visual Analytics Science and Technology, VAST 2017*, pages 1–12, 2018. 37

H. Janetzko, M. Stein, D. Sacha, and T. Schreck. Enhancing parallel coordinates: Statistical visualizations for analyzing soccer data. *Electronic Imaging*, 1:1–8, 2016. 35

S. Jänicke and G. Scheuermann. On the visualization of hierarchical relations and tree structures with tagspheres. In *Proceedings of the Communications in Computer and Information Science*, volume 693, pages 199–219, 2017. 26

S. Jia, P. Lin, Z. Li, J. Zhang, and S. Liu. Visualizing surrogate decision trees of convolutional neural networks. *Journal of Visualization*, 23(1):141–156, 2020. 40

J. Johansson, P. Ljung, M. Jern, and M. Cooper. Revealing structure within clustered parallel coordinates displays. In *Proceedings of the 2005 IEEE Symposium on Information Visualization, InfoVis 2005*, pages 125–132, 2005. 35

J. Johnson, M. Douze, and H. Jegou. Billion-Scale Similarity Search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547, 2021. 102

M. Kahng, P. Y. Andrews, A. Kalro, and D. H. P. Chau. ActiVis: Visual Exploration of Industry-Scale Deep Neural Network Models. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):88–97, 2018. 40

D. Keim, J. Kohlhammer, G. Ellis, and F. Mansmann. *Mastering the information age : solving problems with visual analytics*. Goslar : Eurographics Association, 2010a. 1, 3, 21, 22, 42

D. a. Keim, F. Mansmann, J. Thomas, and D. Keim. Visual Analytics : How Much Visualization and How Much Analytics ? *ACM SIGKDD Explorations Newsletter*, 11(2):5–8, 2010b. 21

J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proceedings of the International Conference on Neural Networks, ICNN 1995*, volume 4, pages 1942–1948 vol.4, 1995. 10

M. Kim, K. Kang, D. Park, J. Choo, and N. Elmqvist. TopicLens: Efficient Multi-Level Visual Topic Exploration of Large-Scale Document Collections. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):151–160, 2017. 14

S. N. Kim, O. Medelyan, M. Y. Kan, and T. Baldwin. SemEval-2010 Task 5: Automatic keyphrase extraction from scientific articles. In *Proceedings of the 5th International Workshop on Semantic Evaluation, ACL 2010 - SemEval 2010*, 2010. 55

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 19

P. Klemm, K. Lawonn, S. Glaßer, U. Niemann, K. Hegenscheid, H. Völzke, and B. Preim. 3D Regression Heat Map Analysis of Population Study Data. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):81–90, 2016. 38

J. Knittel, S. Koch, and T. Ertl. Highlighting Text Regions of Interest with Character-Based LSTM Recurrent Networks. In *Proceedings of the 2018 Postersession at the IEEE Conference on Visualization*, 2018. 41

J. Knittel, S. Koch, and T. Ertl. Interactive Hierarchical Quote Extraction for Content Insights. In J. Madeiras Pereira and R. G. Raidou, editors, *Proceedings of the EuroVis 2019 Posters*. The Eurographics Association, 2019a. 5

J. Knittel, S. Koch, and T. Ertl. Pattern-Based Semantic and Temporal Exploration of Social Media Messages. In *Proceedings of the 2019 IEEE Conference on Visual Analytics Science and Technology, VAST 2019*, pages 134–135, 2019b. 5

J. Knittel, S. Koch, and T. Ertl. PyramidTags: Context-, Time- And Word Order-Aware Tag Maps to Explore Large Document Collections. *IEEE Transactions on Visualization and Computer Graphics*, 27(12):4455–4468, 2021a. 5

J. Knittel, S. Koch, and T. Ertl. ELSKE: Efficient Large-Scale Keyphrase Extraction. In *Proceedings of the 21st ACM Symposium on Document Engineering, DocEng 2021*, New York, NY, USA, 2021b. Association for Computing Machinery. 5

J. Knittel, S. Koch, and T. Ertl. Efficient Sparse Spherical K-Means for Document Clustering. In *Proceedings of the 21st ACM Symposium on Document Engineering, DocEng 2021*, New York, NY, USA, 2021c. Association for Computing Machinery. 5

J. Knittel, A. Lalama, S. Koch, and T. Ertl. Visual Neural Decomposition to Explain Multivariate Data Sets. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1374–1384, 2021d. 6

J. Knittel, S. Koch, T. Tang, W. Chen, Y. Wu, S. Liu, and T. Ertl. Real-Time Visual Analysis of High-Volume Social Media Posts. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):879–889, 2022. 5, 6

M. Krapivin. Large Dataset for Keyphrase Extraction. *Technical Report*, (May 2008), 2008. 55

J. Krause, A. Perer, and E. Bertini. INFUSE: Interactive feature selection for predictive modeling of high dimensional data. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1614–1623, 2014. 38

J. Krause, A. Dasgupta, J. D. Fekete, and E. Bertini. SeekAView: An intelligent dimensionality reduction strategy for navigating high-dimensional data spaces. In *Proceedings of the 2016 IEEE Symposium on Large Data Analysis and Visualization, LDAV 2016*, pages 11–19, 2017. 37

M. Krstajić and D. A. Keim. Visualization of streaming data: Observing change and context in information visualization techniques. In *Proceedings of the 2013 IEEE International Conference on Big Data, Big Data 2013*, pages 41–47, 2013. 3, 31

M. Krstajić, E. Bertini, and D. A. Keim. Cloudlines: Compact display of event episodes in multiple time-series. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2432–2439, 2011. 30

M. Krstajić, M. Najm-Araghi, F. Mansmann, and D. A. Keim. Story tracker: Incremental visual text analytics of news story development. *Information Visualization*, 12(3-4):308–323, 2013. 30

K. Kucher and A. Kerren. Text visualization techniques: Taxonomy, visual survey, and community insights. In *Proceedings of the 2015 IEEE Pacific Visualization Symposium, PacificVis 2015*, pages 117–121, 2015. 23

K. Kucher, R. M. Martins, C. Paradis, and A. Kerren. StanceVis Prime: visual analysis of sentiment and stance in social media texts. *Journal of Visualization*, 23(6):1015–1034, 2020. 31

Z. Kulpa. Diagrammatic Representation of Interval Space in Proving Theorems about Interval Relations. *Reliable Computing*, 3(3):209–217, 1997. 69

K. Lagus, T. Honkela, S. Kaski, and T. Kohonen. Self-Organizing Maps of Document Collections: A New Approach to Interactive Exploration. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, pages 238–243. AAAI Press, 1996. 24

H. Lakkaraju, S. H. Bach, and J. Leskovec. Interpretable Decision Sets: A Joint Framework for Description and Prediction. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 1675–1684, New York, NY, USA, 2016. Association for Computing Machinery. 43

Q. Le and T. Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning, ICML 2014*, volume 4, pages 2931–2939, 2014. 12

B. Lee, N. H. Riche, A. K. Karlson, and S. Carpendale. SparkClouds: Visualizing trends in tag clouds. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1182–1189, 2010. 26, 27

D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999. 13

S. Legg and M. Hutter. A Collection of Definitions of Intelligence. *Frontiers in Artificial Intelligence and applications*, 157:17, 2007. 10

A. Lelu and M. Cadot. Evaluation of Text Clustering Methods and Their Dataspace Embeddings: An Exploration. In *Studies in Classification, Data Analysis, and Knowledge Organization*, volume 5, pages 131–139, 2021. 14, 101

M. Li, T. Zhang, Y. Chen, and A. J. Smola. Efficient Mini-Batch Training for Stochastic Optimization. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 661–670, New York, NY, USA, 2014. Association for Computing Machinery. 19

S. Linnainmaa. The representation of the cumulative rounding error of an algorithm as a Taylor expansion of the local rounding errors. *Master's Thesis (in Finnish), Univ. Helsinki*, pages 6–7, 1970. 19

M. Liu, J. Shi, Z. Li, C. Li, J. Zhu, and S. Liu. Towards Better Analysis of Deep Convolutional Neural Networks. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):91–100, 2017. 40

S. Liu, M. X. Zhou, S. Pan, W. Qian, W. Cai, and X. Lian. Interactive, topic-based visual text summarization and analysis. In *Proceedings of the International Conference on Information and Knowledge Management*, pages 543–552, 2009. 30

S. Liu, X. Wang, Y. Song, and B. Guo. Evolutionary bayesian rose trees. *IEEE Transactions on Knowledge and Data Engineering*, 27(6):1533–1546, 2015a. 15, 31

S. Liu, J. Yin, X. Wang, W. Cui, K. Cao, and J. Pei. Online visual analytics of text streams. *IEEE Transactions on Visualization and Computer Graphics*, 22(11): 2451–2466, 2016. 31

S. Liu, J. Xiao, J. Liu, X. Wang, J. Wu, and J. Zhu. Visual Diagnosis of Tree Boosting Methods. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):163–173, 2018. 39, 44

X. Liu, H. W. Shen, and Y. Hu. Supporting multifaceted viewing of word clouds with focus+context display. *Information Visualization*, 14(2):168–180, 2015b. 25

Y. Liu and R. A. Lopez. The impact of social media conversations on consumer brand choices. *Marketing Letters*, 27(1):1–13, 2016. 23

S. P. Lloyd. Least Squares Quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982. 14, 15, 101

S. Lohmann, J. Ziegler, and L. Tetzlaff. Comparison of tag cloud layouts: Task-related performance and visual exploration. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 5726 LNCS, pages 392–404, 2009. 25

S. Lohmann, F. Heimerl, F. Bopp, M. Burch, and T. Ertl. Concentri cloud: Word cloud visualization for multiple text documents. In *Proceedings of the International Conference on Information Visualisation*, volume 2015-September, pages 114–120, 2015. 27

S. M. Lundberg, B. Nair, M. S. Vavilala, M. Horibe, M. J. Eisses, T. Adams, D. E. Liston, D. K.-W. Low, S.-F. Newman, J. Kim, and S.-I. Lee. Explainable machine-learning predictions for the prevention of hypoxaemia during surgery. *Nature Biomedical Engineering*, 2(10):749–760, 2018. 44

A. M. MacEachren, A. Jaiswal, A. C. Robinson, S. Pezanowski, A. Savelyev, P. Mitra, X. Zhang, and J. Blanford. SensePlace2: GeoTwitter analytics support for situational awareness. In *Proceedings of the 2011 IEEE Symposium on Visual Analytics Science and Technology, VAST 2011*, pages 181–190, 2011. 31

A. Malik, R. Maciejewski, N. Elmqvist, Y. Jang, D. S. Ebert, and W. Huang. A correlative analysis process in a visual analytics environment. In *Proceedings of the 2012 IEEE Conference on Visual Analytics Science and Technology, VAST 2012*, pages 33–42, 2012. 37, 38

C. D. Manning, P. Raghavan, and H. Schutze. *Introduction to Information Retrieval*. Cambridge University Press, 2008. 12

A. Marcus, M. S. Bernstein, O. Badar, D. R. Karger, S. Madden, and R. C. Miller. TwitInfo: Aggregating and visualizing microblogs for event exploration. In *Proceedings of the Conference on Human Factors in Computing Systems*, pages 227–236, 2011. 31

M. Martinc, B. Škrlj, and S. Pollak. TNT-KID: Transformer-based neural tagger for keyword identification, 2021. 55

T. May, A. Bannach, J. Davey, T. Ruppert, and J. Kohlhammer. Guiding feature subset selection with an interactive visualization. In *Proceedings of the 2011 IEEE Symposium on Visual Analytics Science and Technology, VAST 2011*, pages 111–120, 2011. 38

L. McInnes, J. Healy, N. Saul, and L. Großberger. UMAP: Uniform Manifold Approximation and Projection. *Journal of Open Source Software*, 3(29):861, 2018. 16, 17, 35

Q. Mei and C. X. Zhai. Discovering evolutionary theme patterns from text - An exploration of Temporal Text Mining. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 198–207, 2005. 15

R. Meng, S. Zhao, S. Han, D. He, P. Brusilovsky, and Y. Chi. Deep keyphrase generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017*, volume 1, pages 582–592, 2017. 48, 49, 55

R. Mihalcea. Graph-based ranking algorithms for sentence extraction, applied to text summarization. *Proceedings of the ACL 2004 on Interactive Poster and Demonstration Sessions*, pages 20–es, 2004. 48

M. Minsky and S. Papert. *Perceptron: an introduction to computational geometry*. The MIT Press, 1969. 160

S. Mitra, S. A. Seshia, and N. Nicolici. Post-silicon validation opportunities, challenges and recent advances. In *Proceedings of the Design Automation Conference*, pages 12–17, 2010. 138

T. Mühlbacher and H. Piringer. A partition-based framework for building and validating regression models. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):1962–1971, 2013. 38, 39

T. Mühlbacher, L. Linhardt, T. Möller, and H. Piringer. TreePOD: Sensitivity-Aware Selection of Pareto-Optimal Decision Trees. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):174–183, 2018. 39, 43

E. Müller, I. Assent, R. Krieger, T. Jansen, and T. Seidl. Morpheus: Interactive exploration of subspace clustering. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1089–1092, 2008. 37

E. Müller, I. Assent, S. Günnemann, P. Gerwert, M. Hannen, T. Jansen, and T. Seidl. A framework for evaluation and exploration of clustering algorithms in subspaces of high dimensional databases. In *Proceedings of the Lecture Notes in Informatics, LNI*, volume 180, pages 347–366, 2011. 37

M. P. Neto and F. V. Paulovich. Explainable matrix - Visualization for global and local interpretability of random forest classification ensembles. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1427–1437, 2021. 39, 137

T. D. Nguyen and M.-Y. Kan. Keyphrase Extraction in Scientific Publications. In D. H.-L. Goh, T. H. Cao, I. T. Sølvberg, and E. Rasmussen, editors, *Asian Digital Libraries. Looking Back 10 Years and Forging New Frontiers*, pages 317–326, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg. 55

N. J. Nilsson. *Introduction to Machine Learning - an early draft of a proposed textbook*. 2005. 10

N. J. Nilsson. *The quest for artificial intelligence*. Cambridge University Press, 2009. 10

M. Novotný and H. Hauser. Outlier-preserving focus+context visualization in parallel coordinates. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):893–900, 2006. 35

N. Oliveira, P. Cortez, and N. Areal. The impact of microblogging data for stock market prediction: Using Twitter to predict returns, volatility, trading volume and survey sentiment indices. *Expert Systems with Applications*, 73:125–144, 2017. 23

S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010. 41

D. Park, S. Kim, J. Lee, J. Choo, N. Diakopoulos, and N. Elmqvist. ConceptVector: Text Visual Analytics via Interactive Lexicon Building Using Word Embedding. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):361–370, 2018. 29

C. L. Paul, J. Chang, A. Endert, N. Cramer, D. Gillen, S. Hampton, R. Burtner, R. Perko, and K. A. Cook. TexTonic: Interactive visualization for exploration and discovery of very large text collections. *Information Visualization*, 18(3): 339–356, 2019. 26

F. V. Paulovich and R. Minghim. Text Map Explorer: A tool to create and explore document maps. In *Proceedings of the International Conference on Information Visualisation*, pages 245–251, 2006. 24

F. V. Paulovich, L. G. Nonato, R. Minghim, and H. Levkowitz. Least square projection: A fast high-precision multidimensional projection technique and its application to document mapping. *IEEE Transactions on Visualization and Computer Graphics*, 14(3):564–575, 2008. 24

F. V. Paulovich, F. M. Toledo, G. P. Telles, R. Minghim, and L. G. Nonato. Semantic wordification of document collections. *Computer Graphics Forum*, 31 (3 PART 3):1145–1153, 2012. 24

M. Peng, J. Zhu, H. Wang, X. Li, Y. Zhang, X. Zhang, and G. Tian. Mining event-oriented topics in microblog stream with unsupervised multi-view hierarchical embedding. *ACM Transactions on Knowledge Discovery from Data*, 12(3), 2018. 15

G. Piatetsky-Shapiro. Knowledge Discovery in Real Databases: A Report on the IJCAI-89 Workshop. *AI Magazine*, 11(4):68, 1990. 19, 22

H. Piringer, W. Berger, and H. Hauser. Quantifying and comparing features in high-dimensional datasets. In *Proceedings of the International Conference on Information Visualisation*, pages 240–245, 2008. 38

R. Poli, J. Kennedy, and T. Blackwell. Particle swarm optimization. *Swarm Intelligence*, 1(1):33–57, 2007. 11

A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. [GPT-2] Language Models are Unsupervised Multitask Learners. *OpenAI Blog*, 1(May): 1–7, 2020. 12, 17

M. R. H. Rakib, N. Zeh, and E. Milios. Efficient Clustering of Short Text Streams Using Online-Offline Clustering. In *Proceedings of the 21st ACM Symposium on Document Engineering, DocEng 2021*, New York, NY, USA, 2021. Association for Computing Machinery. 16

E. Ramos and D. Donoho. ASA Data Exposition dataset, 1983. [Online]. Available: `http://stat-computing.org/dataexpo/1983.html`. 147

T. Raykov and G. A. Marcoulides. *An Introduction to applied multivariate analysis*. Routledge, 2008. 34

G. Richer, J. Sansen, F. Lalanne, D. Auber, and R. Bourqui. Enabling hierarchical exploration for large-scale multidimensional data with abstract parallel coordinates. In *Proceedings of the CEUR Workshop*, volume 2083, pages 76–83, 2018. 35

A. W. Rivadeneira, D. M. Gruen, M. J. Muller, and D. R. Millen. Getting our head in the clouds: Toward evaluation studies of tagclouds. In *Proceedings of the Conference on Human Factors in Computing Systems*, pages 995–998, 2007. 68

C. Rohrdantz, D. Oelke, M. Krstajić, and F. Fischer. Real-Time Visualization of Streaming Text Data : Tasks and Challenges. In *Proceedings of the VIS-Week 2011 Workshops*, 2011. 3, 31

S. Rose, D. Engel, N. Cramer, and W. Cowley. Automatic Keyword Extraction from Individual Documents. In *Text Mining: Applications and Theory*, pages 1–20. John Wiley and Sons, 2010. 48

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986. 19

D. Sacha, M. Kraus, D. A. Keim, and M. Chen. VIS4ML: An Ontology for Visual Analytics Assisted Machine Learning. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):385–395, 2019. 40

G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523, 1988. 13

C. Sammut and G. I. Webb. *Encyclopedia of Machine Learning*. Springer Science & Business Media, 2010. 10

T. Y. S. S. Santosh, D. K. Sanyal, P. K. Bhowmick, and P. P. Das. DAKE: Document-level attention for keyphrase extraction. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 12036 LNCS, pages 392–401, 2020. 48

V. M. Savage, A. P. Allen, J. H. Brown, J. F. Gillooly, A. B. Herman, W. H. Woodruff, and G. B. West. Scaling of number, size, and metabolic rate of cells with body size in mammals. *Proceedings of the National Academy of Sciences*, 104(11):4718–4723, 2007. 9

M. Scherer, J. Bernard, and T. Schreck. Retrieval and exploratory search in multivariate research data repositories using regressional features. In *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries*, pages 363–372, 2011. 37

A. Schneider, G. Hommel, and M. Blettner. Lineare Regressionsanalyse. *Deutsches Arzteblatt*, 107(44):776–782, 2010. 45

J. Schrammel, M. Leitner, and M. Tscheligi. Semantically structured tag clouds: An empirical evaluation of clustered presentation approaches. In *Proceedings of the Conference on Human Factors in Computing Systems*, pages 2037–2040, 2009. 25

M. Sedlmair, M. Brehmer, S. Ingram, and T. Munzner. Dimensionality reduction in the wild: gaps and guidance. *Dept. Comput. Sci., Univ. British Columbia, Vancouver, BC, Canada, Tech. Rep. TR-2012-03*, 2012. 36

M. Sedlmair, C. Heinzl, S. Bruckner, H. Piringer, and T. Moller. Visual parameter space analysis: A conceptual framework. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2161–2170, 2014. 136

C. Seifert, B. Kump, W. Kienreich, G. Granitzer, and M. Granitzer. On the beauty and usability of tag clouds. In *Proceedings of the International Conference on Information Visualisation*, pages 17–25, 2008. 25

Y. Shi and R. Eberhart. A modified particle swarm optimizer. In *Proceedings of the 1998 IEEE International Conference on Evolutionary Computation*, pages 69–73, 2002. 10, 83

J. Sinclair and M. Cardew-Hall. The folksonomy tag cloud: When is it useful? *Journal of Information Science*, 34(1):15–29, 2008. 25, 68

B. Škrlj, A. Repar, and S. Pollak. RaKUn: Rank-based Keyword Extraction via Unsupervised Learning and Meta Vertex Aggregation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 11816 LNAI, pages 311–323, 2019. 48

J. Stasko, C. Görg, and Z. Liu. Jigsaw: Supporting investigative analysis through interactive visualization. *Information Visualization*, 7(2):118–132, 2008. 28

E. Steiger, B. Resch, and A. Zipf. Exploration of spatiotemporal and semantic clusters of Twitter data using unsupervised neural networks. *International Journal of Geographical Information Science*, 30(9):1694–1716, 2016. 31

G. Stilo and P. Velardi. Efficient temporal mining of micro-blog texts and its application to event discovery. *Data Mining and Knowledge Discovery*, 30(2): 372–402, 2016. 15

H. Strobelt, S. Gehrmann, H. Pfister, and A. M. Rush. LSTMVis: A Tool for Visual Analysis of Hidden State Dynamics in Recurrent Neural Networks. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):667–676, 2018. 41

H. K. Sul, A. R. Dennis, and L. I. Yuan. Trading on Twitter: Using Social Media Sentiment to Predict Stock Returns. *Decision Sciences*, 48(3):454–488, 2017. 23

G. Sun, Y. Wu, S. Liu, T. Q. Peng, J. J. Zhu, and R. Liang. EvoRiver: Visual analysis of topic coopetition on social media. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1753–1762, 2014. 30

A. A. Talin, F. Léonard, A. M. Katzenmeyer, B. S. Swartzentruber, S. T. Picraux, M. E. Toimil-Molares, J. G. Cederberg, X. Wang, S. D. Hersee, and A. Rishinaramangalum. Transport characterization in nanowires using an electrical nanoprobe. *Semiconductor Science and Technology*, 25(2), 2010. 26

A. Tatu, G. Albuquerque, M. Eisemann, J. Schneidewind, H. Theisel, M. Magnork, and D. Keim. Combining automated analysis and visualization techniques for effective exploration of high-dimensional data. In *Proceedings of the 2009 IEEE Symposium on Visual Analytics Science and Technology, VAST 2009*, pages 59–66, 2009. 38

A. Tatu, F. Maaß, I. Färber, E. Bertini, T. Schreck, T. Seidl, and D. Keim. Subspace search and visualization to make sense of alternative clusterings in high-dimensional data. In *Proceedings of the 2012 IEEE Conference on Visual Analytics Science and Technology, VAST 2012*, pages 63–72, 2012a. 37

A. Tatu, L. Zhang, E. Bertini, T. Schreck, D. Keim, S. Bremm, and T. Von Landesberger. ClustNails: Visual analysis of subspace clusters. *Tsinghua Science and Technology*, 17(4):419–428, 2012b. 37

C. Teflioudi and R. Gemulla. Exact and approximate maximum inner product search with LEMP. *ACM Transactions on Database Systems*, 42(1), 2016. 102

D. Thom, H. Bosch, S. Koch, M. Worner, and T. Ertl. Spatiotemporal anomaly detection through visual analysis of geolocated Twitter messages. In *Proceedings of the 2012 IEEE Pacific Visualization Symposium, PacificVis 2012*, pages 41–48, 2012. 33

D. Thom, R. Kruger, T. Ertl, U. Bechstedt, A. Platz, J. Zisgen, and B. Volland. Can twitter really save your life? A case study of visual social media analytics for situation awareness. In *Proceedings of the 2015 IEEE Pacific Visualization Symposium, PacificVis 2015*, pages 183–190, 2015. 23, 33

J. J. Thomas and K. A. Cook. *Illuminating the path: The research and development agenda for visual analytics*. 2005. 1, 3, 21

M. E. Tipping and C. M. Bishop. Mixtures of Probabilistic Principal Component Analyzers. *Neural Computation*, 11(2):443–482, 1999. 16, 35

G. Tkachev, S. Frey, and T. Ertl. Local Prediction Models for Spatiotemporal Volume Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 27(7):3091–3108, 2021. 41

T. Tompson and J. Benz. AP VoteCast 2018, 2019. [Online]. Available: `http://doi.org/10.3886/E109687V2`. 163

C. Turkay, A. Lundervold, A. J. Lundervold, and H. Hauser. Representative factor generation for the interactive visual analysis of high-dimensional data. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2621–2630, 2012. 37

N. Van de Weghe, R. Docter, P. De Maeyer, B. Bechtold, and K. Ryckbosch. The triangular model as an instrument for visualising and analysing residuality. *Journal of Archaeological Science*, 34(4):649–655, 2007. 69

S. Van Den Elzen and J. J. Van Wijk. BaobabView: Interactive construction and analysis of decision trees. In *Proceedings of the 2011 IEEE Symposium on Visual Analytics Science and Technology, VAST 2011*, pages 151–160, 2011. 39, 40

L. Van Der Maaten and G. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(11):2579–2605, 2008. 16, 35

F. Van Ham, M. Wattenberg, and F. B. Viégas. Mapping text with phrase nets. In *IEEE Transactions on Visualization and Computer Graphics*, volume 15, pages 1169–1176, 2009. 28, 57

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information*

Processing Systems, volume 2017-December of *NIPS'17*, pages 5999–6009, Red Hook, NY, USA, 2017. Curran Associates Inc. 17

F. Viégas, M. Wattenberg, J. Hebert, G. Borggaard, A. Cichowlas, J. Feinberg, J. Orwant, and C. R. Wren. Google+ Ripples: A native visualization of information flow. In *Proceedings of the 22nd International Conference on World Wide Web, WWW 2013*, pages 1389–1398, 2013. 31

F. B. Viégas and M. Wattenberg. TIMELINES: Tag clouds and the case for vernacular visualization. *Interactions*, 15(4):49, 2008. 25, 68

F. B. Viégas, M. Wattenberg, and J. Feinberg. Participatory visualization with wordle. In *IEEE Transactions on Visualization and Computer Graphics*, volume 15, pages 1137–1144, 2009. 25

X. Wan and J. Xiao. Single document keyphrase extraction using neighborhood knowledge. In *Proceedings of the National Conference on Artificial Intelligence*, volume 2, pages 855–860, 2008. 48

B. Wang and K. Mueller. The Subspace Voyager: Exploring High-Dimensional Data along a Continuum of Salient 3D Subspaces. *IEEE Transactions on Visualization and Computer Graphics*, 24(2):1204–1222, 2018. 37

D. Wang, D. Tan, and L. Liu. Particle swarm optimization algorithm: an overview. *Soft Computing*, 22(2):387–408, 2018. 11

J. Wang, J. Zhao, S. Guo, C. North, and N. Ramakrishnan. ReCloud: Semantics-based word cloud visualization of user reviews. In *Proceedings of the Graphics Interface*, pages 151–158, 2014. 25, 26, 68, 70

Q. Wang, Z. Chen, Y. Wang, and H. Qu. A Survey on ML4VIS: Applying MachineLearning Advances to Data Visualization. *IEEE Transactions on Visualization and Computer Graphics*, page 1, 2021. 41

X. Wang and A. McCallum. Topics over Time: A non-markov continuous-time model of topical trends. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, volume 2006, pages 424–433, 2006. 15

X. Wang, W. Dou, Z. Ma, J. Villalobos, Y. Chen, T. Kraft, and W. Ribarsky. I-SI: Scalable architecture for analyzing latent topical-level information from social media data. *Computer Graphics Forum*, 31(3 PART 4):1275–1284, 2012. 30

X. Wang, S. Liu, Y. Song, and B. Guo. Mining evolutionary multi-branch trees from text streams. In *Proceedings of the ACM SIGKDD International Conference* *on Knowledge Discovery and Data Mining*, volume Part F128815, pages 722–730, 2013. 15

X. Wang, S. Liu, J. Liu, J. Chen, J. Zhu, and B. Guo. TopicPanorama: A Full Picture of Relevant Topics. *IEEE Transactions on Visualization and Computer Graphics*, 22(12):2508–2521, 2016. 28

Y. Wang, J. Li, H. P. Chan, I. King, M. R. Lyu, and S. Shi. Topic-aware neural keyphrase generation for social media language. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, ACL 2019*, pages 2516–2526, 2020. 48

M. Wattenberg and F. B. Viégas. The word tree, an interactive visual concordance. In *IEEE Transactions on Visualization and Computer Graphics*, volume 14, pages 1221–1228, 2008. 27, 57

J. J. Webster and C. Kit. Tokenization as the Initial Phase in NLP. In *Proceedings of the 14th Conference on Computational Linguistics - Volume 4*, COLING '92, pages 1106–1110, USA, 1992. Association for Computational Linguistics. 12

E. Weippl. Visualizing content based relations in texts. In *Proceedings Second Australasian User Interface Conference. AUIC 2001*, pages 34–41, 2001. 24

J. A. Wise, J. J. Thomas, K. Pennock, D. Lantrip, M. Pottier, A. Schur, and V. Crow. Visualizing the non-visual: spatial analysis and interaction with information from text documents. In *Proceedings of the Information Visualization Conference*, pages 51–58, 1995. 24

P. C. Wong. Visual Data Mining. *IEEE Computer Graphics and Applications*, 19(5): 20–21, 1999. 20, 22

K. Wongsuphasawat, D. Moritz, A. Anand, J. Mackinlay, B. Howe, and J. Heer. Voyager: Exploratory Analysis via Faceted Browsing of Visualization Recommendations. *IEEE Transactions on Visualization and Computer Graphics*, 22(1): 649–658, 2016. 37

K. Wongsuphasawat, Z. Qu, D. Moritz, R. Chang, F. Ouk, A. Anand, J. MacKinlay, B. Howe, and J. Heer. Voyager 2: Augmenting visual analysis with partial view specifications. In *Proceedings of the Conference on Human Factors in Computing Systems*, pages 2648–2659, 2017. 37

Y. Wu, T. Provan, F. Wei, S. Liu, and K. L. Ma. Semantic-preservingword clouds by seam carving. *Computer Graphics Forum*, 30(3):741–750, 2011. 26, 68

Y. Wu, S. Liu, K. Yan, M. Liu, and F. Wu. OpinionFlow: Visual analysis of opinion diffusion on social media. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1763–1772, 2014. 31

Y. Wu, N. Cao, D. Gotz, Y. P. Tan, and D. A. Keim. A Survey on Visual Analytics of Social Media Data. *IEEE Transactions on Multimedia*, 18(11):2135–2148, 2016. 30

Y. Wu, Z. Chen, G. Sun, X. Xie, N. Cao, S. Liu, and W. Cui. StreamExplorer: A Multi-Stage System for Visually Exploring Events in Social Streams. *IEEE Transactions on Visualization and Computer Graphics*, 24(10):2758–2772, 2018. 25, 33, 116

L. Xiong, C. Hu, C. Xiong, D. Campos, and A. Overwijk. Open domain web keyphrase extraction beyond language modeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019*, pages 5175–5184, 2020. 48

J. Xu, Y. Tao, and H. Lin. Semantic word cloud generation based on word embeddings. In *Proceedings of the 2016 IEEE Pacific Visualization Symposium, PacificVis 2016*, pages 239–243, 2016. 26

W. Xu, X. Liu, and Y. Gong. Document Clustering Based On Non-negative Matrix Factorization. In *Proceedings of the SIGIR Forum (ACM Special Interest Group on Information Retrieval)*, number SPEC. ISS., pages 267–273, 2003. 13

W. Yang, X. Wang, J. Lu, W. Dou, and S. Liu. Interactive Steering of Hierarchical Clustering. *IEEE Transactions on Visualization and Computer Graphics*, 27(10): 3953–3967, 2021. 29

H. Ye and L. Wang. Semi-supervised learning for neural keyphrase generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018*, pages 4142–4153, 2020. 48

S. M. Yimam, H. Ulrich, T. von Landesberger, M. Rosenbach, M. Regneri, A. Panchenko, F. Lehmann, U. Fahrer, C. Biemann, and K. Ballweg. new/s/leak – Information Extraction and Visualization for Investigative Data Journalists. In *Proceedings of ACL 2016 System Demonstrations*, pages 163–168, 2016. 23, 67

J. Yin, D. Chao, Z. Liu, W. Zhang, X. Yu, and J. Wang. Model-Based Clustering of Short Text Streams. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery And Data Mining*, KDD '18, pages 2634–2642, New York, NY, USA, 2018. Association for Computing Machinery. 16

C. Zhang, J. Yang, F. B. Zhan, X. Gong, J. D. Brender, P. H. Langlois, S. Barlowe, and Y. Zhao. A visual analytics approach to high-dimensional logistic regression modeling and its application to an environmental health study. In *Proceedings of the 2016 IEEE Pacific Visualization Symposium, PacificVis 2016*, pages 136–143, 2016. 39

J. Zhang, Y. Song, C. Zhang, and S. Liu. Evolutionary hierarchical Dirichlet processes for multiple correlated time-varying corpora. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1079–1088, 2010. 15

J. Zhang, C. Surakitbanharn, N. Elmqvist, R. Maciejewski, Z. Qian, and D. S. Ebert. TopoText: Context-preserving Text data exploration across multiple spatial scales. In *Proceedings of the Conference on Human Factors in Computing Systems*, pages 1–13, 2018. 31

Z. Zhang, K. T. McDonnell, E. Zadok, and K. Mueller. Visual correlation analysis of numerical and categorical data on the correlation map. *IEEE Transactions on Visualization and Computer Graphics*, 21(2):289–303, 2015. 38

S. Zhong. Efficient streaming text clustering. *Neural Networks*, 18(5):790–798, 2005. 16

L. M. Zintgraf, T. S. Cohen, T. Adel, and M. Welling. Visualizing deep neural network decisions: Prediction difference analysis. In *Proceedings of the 5th International Conference on Learning Representations, ICLR 2017*, 2017. 40