Institut für Maschinelle Sprachverarbeitung
Universität Stuttgart
Pfaffenwaldring 5B
D-70569 Stuttgart

Bachelorarbeit

Optimizing Human Annotation of Word Usage Graphs in a Realistic Simulation Environment

Serge Kotchourko

Studiengang:	B.Sc. Informatik
Prüfer:	Apl. Prof. Dr. Sabine Schulte im Walde
Betreuer:	Dominik Schlechtweg
Beginn der Arbeit:	01.05.2021
Ende der Arbeit:	01.11.2021

Contents

Abstract

Word Usage Graphs (WUGs) are an approach of representing relations between word usage pairs, where each word usage is considered as a node and the weighted undirected edge between such a pair represents its semantic proximity. This shifts problems of Computational Linguistics into the graph problem space. There is only little research into how such WUGs can be annotated efficiently and effectively. Therefore, we build a simulation to test a broad range of sampling, clustering and stopping procedures with respect to their impact on finding good solutions. We show that it is possible to simulate graphs which share characteristics close to the observed WUGs. Based on this we are able to scrutinize various annotation procedures and are able to extract their advantages and disadvantages for the annotation process.

Wortverwendungsgraphen (WUGs) sind ein Ansatz zur Darstellung von Beziehungen zwischen Wortverwendungspaaren, wobei jede Wortverwendung als Knoten dargestellt wird und die gewichtete ungerichtete Kante zwischen einem solchen Paar die semantische Nähe darstellt. Somit können Probleme der Computerlinguistik in den Bereich der Graphen verlagert werden. Es gibt nur wenig Forschung darüber, wie solche WUGs effizient und effektiv annotiert werden können. Daher entwickeln wir eine Simulation, um eine breite Auswahl an Sampling, Cluster und Stop-Prozeduren hinsichtlich ihrer Auswirkungen auf das Finden guter Lösungen zu testen. Wir zeigen, dass es möglich ist, Graphen zu simulieren, deren Eigenschaften nah der beobachteten WUGs ähnelt. Auf dieser Grundlage sind wir in der Lage, verschiedene Annotationsverfahren zu untersuchen und ihre Vor- und Nachteile für den Annotationsprozess zu extrahieren.

1 Introduction

Word Usage Graphs (WUGs, McCarthy et al., 2016; Schlechtweg et al., 2021b) is an approach of representing relations between word usage pairs, where each word usage is considered as a node and the weighted undirected edge between such a pair represents its semantic proximity (Find example WUGs in Figure 1). This representation is convenient, as it shifts the initial problem, like semantic change detection (Schlechtweg et al., 2020) or sense detection (McCarthy et al., 2016), into a graph theory problem space, allowing for new approaches to be defined and opens the possibility of translating other well researched methods to this research area.

This representation does bear its drawbacks. For sense detection a fully annotated graph would be the ideal, but the number of edges to be annotated grows quadraticaly with number of nodes present in a WUG. Considering that each annotations has to be done by a human annotator makes this an infeasible goal. Hence, it is crucial to find approaches which solve this problem effectively and efficiently. Efficiency is needed to reduce the annotation load, while effectiveness is needed to find word usage pairs which carry the most information, such that a correct sense structure can be found. These approaches also need to account for human annotators, which can introduce noise because of ambiguity, unknown context (Schlechtweg et al., 2021b) or when non-expert annotators are utilized (Schlechtweg et al., 2018). Thus they need to be robust against effects arising from the annotation process itself.

Our aim is to find such efficient and effective models, consisting of a sampling and clustering strategy as well as a stopping criterion. This is done by evaluating models in a simulation environment, which uses a generative procedure to generate complete WUGs (*True WUGs*), which are assumed to be the ground truth, and simulate the complete annotation procedure based on this ground truth and chosen model, thus generating *Annotated WUGs*. These *Annotated WUGs* are then evaluated on their correspondence respectively to their *True WUGs*, resulting in an assessment for the models effectiveness and efficiency.

Section 3 describes the data used in the generative process of *True WUGs*. In

Section 4 we take a look at some important characteristics of WUGs from the DWUG DE/EN and DiscoWUG data set, and reviewed different approaches used during the simulation in Section 5 and Section 6. Finally, we will discuss benefits and drawbacks of each approach and what the best performing models are (Section 7).

2 Related Work

WUGs are a relatively new model of representing graded word usage relatedness (Erk et al., 2013; McCarthy et al., 2016; Schlechtweg et al., 2021b; Kurtyigit et al., 2021a). McCarthy et al. (2016) exploited the graph representation to research the extend of clustering strategies as a tool for partitioning graphs into sense clusters. Previous simulation done by (Schlechtweg et al., 2020; 2021b) introduced models on how word usages could be sampled for the annotation process and how they could be clustered. Yet there are no studies on how different sampling and clustering strategies may affect the resulting WUGs and thus sense discovery. Such studies would also take a long time and careful planing due to the need of human annotators.

Schlechtweg et al. (2021a) recently showed in his study, that by using a simple generative model, the Weighted Stochastic Block Model (Aicher et al., 2014; Peixoto, 2017), an extension to the Stochastic Block Model (Holland et al., 1983), it is possible to model Word Usage Graphs. Considering recent studies on creating graded annotated data sets done by Schlechtweg et al. (2021b), building a large data set with 100,000 human annotations for English, German, Swedish and Latin, DiscoWUG by Kurtyigit et al. (2021a), DURel Schlechtweg et al. (2018) and others, yielded a wide variety of WUGs. This opens the possibility to generate Word Usage Graphs without the need for human annotators, as these data sets can be used for the generative model. Hence, allowing the research and comparison of possible models for an more efficient annotation process and better sense detection.

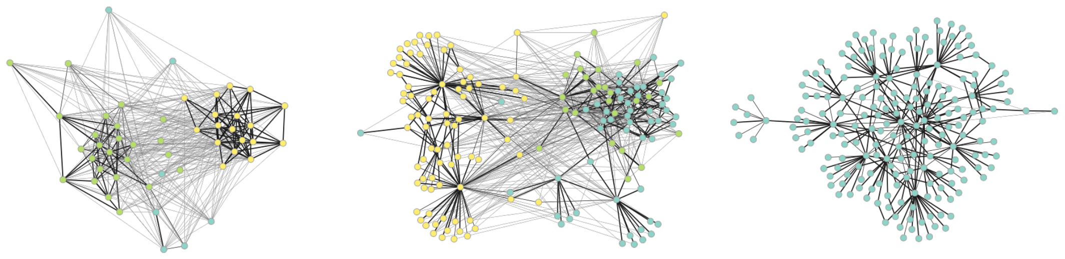

Figure 1: WUGs of *Zehner* from DiscoWUG (left), *abbauen* from DWUG DE (middle) and *bag* from DWUG EN (right). Nodes represent usages of the respective target word. A nodes color represents its membership to a cluster. Edge-weights represent the median of relatedness annotations between usages, where **black**/gray edge color represents **high**/low edge-weights, i.e., weights ≥ 2.5/weights < 2.5.

3 Data

We utilize the annotated English, German DWUG (DWUG DE/EN, Version 1.0.0) (Schlechtweg et al., 2021b) and DiscoWUG (Version 1.0.0) (Kurtyigit et al., 2021a) data sets, where each WUG represents a target word and contains a set of word usages (nodes) from two time periods.[1] Word usage pairs are connected by a weighted edge, representing their semantic relatedness, which is the median of all annotations for this edge. The possible annotation are chosen from the DURel relatedness scale (Schlechtweg et al., 2018) and can be found in Table 1. An edge which is annotated with a zero, meaning that the annotator can not decide on the semantic relatedness of a pair, is not included in the calculation of the weight. Figure 1 visualizes three examples of WUGs from the chosen data sets.

We choose these data sets due to a number of reasons. DWUG DE/EN is interesting, as it contains large WUGs with a lot of annotations where usages were randomly sampled from a real corpus. They are especially interesting, as these are two different languages, which might show different characteristics, thus models may differ significantly in their performance. DiscoWUG is also interesting, as its word usages were also randomly sampled from a real corpus, but noisy words were ex-

[1] `https://www.ims.uni-stuttgart.de/data/wugs`

4: Identical
↑ 3: Closely Related
2: Distantly Related
1: Unrelated

0: Cannot decide

Table 1: DURel relatedness scale.

cluded. Furthermore, due to its small size and edges between word usages being randomly sampled per annotator, we suspect that less bias was introduced and that the edge-weight distribution resembles the true edge-weight distribution closely.

As we want the simulated annotation process to mirror the real process as closely as possible, sampling from a small human annotated set is not possible. This is why we used the generative model introduced by Schlechtweg et al. (2021a) to generate the following two set.

Fitted WUGs This set of complete WUGs are generated based on the DWUG DE/EN and DiscoWUG data set, and were generated by the process described by Schlechtweg et al. (2021a). A more in-depth explanation on how we generated these can be found in Section 6.

Coarse WUGs *Coarse WUGs* are based on observed characteristics of DWUG DE/EN and DiscoWUG. We use these as parameters for the generative process. Which observations were made can be found in Section 4, and how this data is utilized to construct *Coarse WUGs* can be found in Section 6.

4 Word Usage Graph Analysis

Understanding the current state of Word Usage Graphs is imperative for designing both efficient and effective approaches, but especially for designing a useful simulation suit, as we take advantage of human annotated WUGs to generate closely related WUGs. For this we need to take a look at different characteristics emerging from the currently annotated data sets.

Kilgarriff (1997) argues that the frequency of a word used in a specific context leads to its observed senses. Examining the characteristics of sense distribution and sizes may provide helpful approaches, as capturing these characteristic would be crucial in developing reasonable artificial WUGs, but also would be an important criteria in developing reasonable models. Hence in Section 4.2 we will take a closer look at how senses are distributed and their sense size.

For word usage pairs with the same sense, we would expect that they would score a high ranking on a graded relatedness scale, as described by Erk et al. (2013) 5-point graded scale or DURels relatedness scale by Schlechtweg et al. (2018) (Table 1), but due to ambiguity, unfamiliar meaning, context (Schlechtweg et al., 2021b) or non-expert annotations Schlechtweg et al. (2018), these word pairs may be annotated with an high annotator disagreement. Therefore, evaluating the current state of how word usage pairs are annotated is important for choosing, building or refining clustering approaches for WUGs (Section 4.3). Considering this, understanding the current state of Word Usage Graphs is imperative for designing efficient and effective approaches, but especially for designing a useful simulation suit.

4.1 Word Usages

Before we take a look at senses and annotations we need to evaluate the feasibility of annotating a WUG fully, since if it is possible to annotate a WUG fully, there is no reasonable explanation in employing sampling strategies.

$$|E| = \frac{|N|(|N|-1)}{2} = \frac{|N|^2 - |N|}{2} \tag{1}$$

The number of word usages $|N|$ directly correlate with feasibility of annotating a WUG fully, as the number of edges (word usage pairs, $|E|$) grows quadratic with the number of nodes (word usages) (Equation 1). Hence, fully annotating WUGs, where the number of nodes are small (e.g. $|N| = 10 \rightarrow |E| = 45$) is feasible, while WUGs

	$\lvert\bar{N}\rvert$	$\lvert\tilde{N}\rvert$	$\min\lvert N\rvert$	$\max\lvert N\rvert$	$\lvert\bar{E}\rvert$	$\lvert\tilde{E}\rvert$	$\min\lvert E\rvert$	$\max\lvert E\rvert$
DWUG DE	176	183	126	200	15305	16562	7875	19900
DWUG EN	193	197	165	200	18559	19306	13530	19900
DiscoWUG	49	49	45	50	1175	1176	990	1225
All	120	149	45	200	7172	10952	990	19900

Table 2: Mean, Median, Minimum and Maximum number of nodes per WUG in DWUG DE/EN and DiscoWUG and the resulting number of edges for fully annotating a WUG.

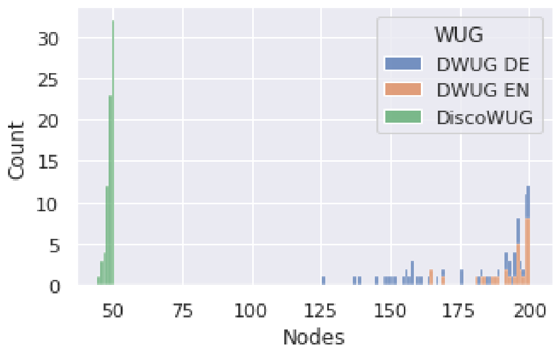

Figure 2: Histogram of nodes per WUG in DWUG DE/EN and DiscoWUG.

with $|N| = 50$ ($\rightarrow |E| = 1225$) may already be outside the scope of possibility. In such cases, only a subset of edges can be annotated. The number of nodes in a WUG for the chosen data set can be seen in Figure 2 and Table 2 shows that even for the smallest WUGs, with only 45 nodes, the annotation process may already be outside the possible realm, especially if we want to include multiple annotations for cases where word usages show ambiguity, word usage pairs which are annotated with zeros on the DURel scale (Table 1) or have a high annotator disagreement. Therefore it is imperative that edges are selected on some bases to ensure that they provide the most information possible.

	$\lvert\bar{C}\rvert$	$\lvert\tilde{C}\rvert$	$\lvert\tilde{C_0}\rvert$	$\lvert\tilde{C_1}\rvert$	$\lvert\tilde{C_2}\rvert$	$\lvert\tilde{C}_{i\geq 3}\rvert$
DWUG DE	5.7	4.5	.84	.09	.01	.00
DWUG EN	8.2	5.0	.92	.05	.02	.00
DiscoWUG	4.8	4.0	.77	.12	.02	.00
All	5.9	5.0	.85	.08	.02	.00

Table 3: Number of senses and relative size of senses in DWUG DE/EN and DiscoWUG.

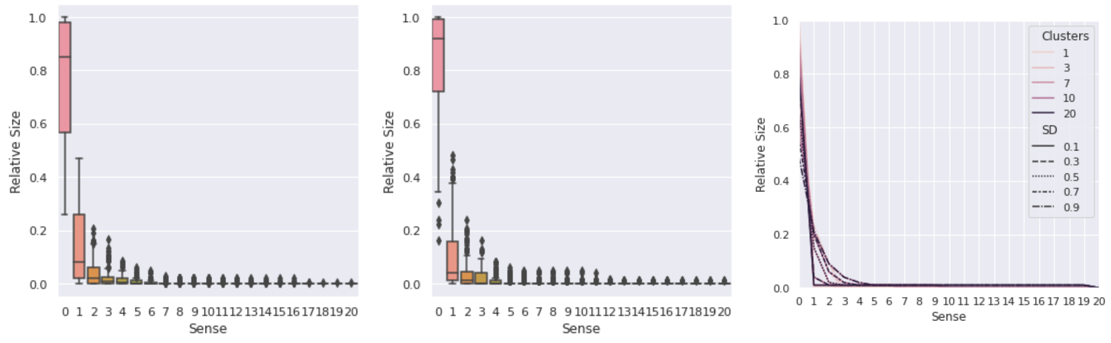

Figure 3: **Left:** Relative sense size for all WUGs of the data set. **Middle:** Relative sense size where word usages were separated by their respective time-period. **Right:** Comparison to a Log-normal distribution.

4.2 Senses

As mentioned above, understanding sense structure is important in reasoning how to build WUGs that resemble real world data, but even more importantly for finding good approaches that capture the underlying characteristics of a words senses. For this we will take a look at two characteristics, the number of senses a WUG displays and its sense sizes. Understanding these characteristics is especially interesting, as they may be the main factor for choosing reasonable sampling and clustering strategies.

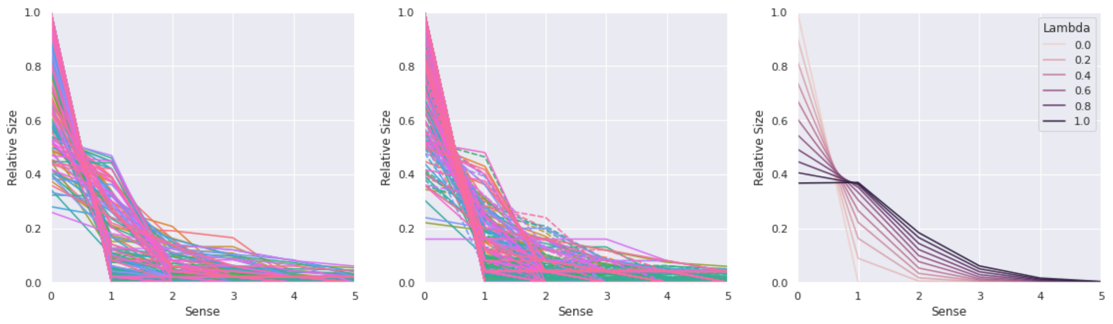

Figure 4: **Left:** Relative sense size for single WUGs in the data set. **Middle:** Relative sense size for single WUGs, where word usages were separated by their respective time-period. **Right:** Comparison to Poisson-distribution with different λ values.

Sense Sizes Figure 3 shows that the first sense is mostly dominant, while following senses are less prominent. This is in line with observations made by Kilgarriff (1997), which argues that the first sense is the most frequently observed sense for word usages and that following senses rapidly decline in number of observed word usages. If we split the word usages based on their time period, as our data set consists of word usages from two time periods, this effect seems to be even more prominent. In general it seems that the sense distribution follows a log-normal distribution.

Figure 4 shows the relative sense size for each WUG separately. Comparing this against a Poisson-Distribution, we observe that each WUGs relative sense size actually follows this distribution more closely. Still we can observe, that the most dominant sense has the most word usages associated with.

Number of Senses We observe that most WUGs have between 1 and 5 senses (Figure 5 and Table 3) and that WUGs with high number of senses are unlikely and only occur rarely. Comparing the number of senses and their relative sizes (see Table 3), it shows that with the rise of senses for a word, the number of word usages associated with these additional senses tend to be very small. For example, if we had a WUG with 100 word usages and 3 senses, we would expect only around 2 word usages to express this third sense. Finding this sense would mean a lot of work, as only around 4% of edges express this relationship and a sufficient number of these

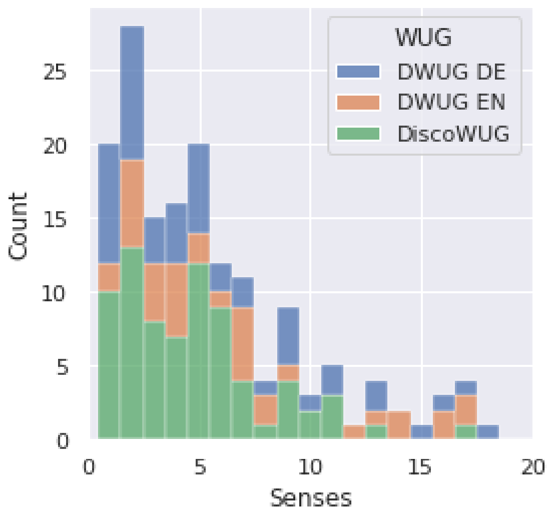

Figure 5: Number of senses per WUG in DWUG DE/EN and DiscoWUG.

have to be sampled to identify this sense correctly.

4.3 Annotation

As we want to simulate a realistic annotation process, it is important to analyze how annotations are distributed and how annotators differ compared to each other. The number of annotations per WUG is an interesting property to consider, as it provides a point at which a model should exhibit good performance.

Number of Annotations Most WUGs have around 500 annotations, see Figure 6 (left). For even the smallest WUGs, which are found in the data set DiscoWUG, with around 49 word usages (Table 2) and only around 300 annotations, only around 25% percent of word usage pairs are annotated, which is even lower for the DWUG DE/EN data set with only around 3.5% of word usage pairs being annotated. With such a low number of word usage pairs being annotated in the data set, it shows the importance of choosing the pairs carefully, such that the characteristics of a word can be captured efficiently.

	$\|\bar{A}\|$	$\|\tilde{A}\|$	$\frac{\|A_0\|}{\|A\|}$	$\bar{\mathrm{Var}}A_E$	$\tilde{\mathrm{Var}}A_E$	ρ_{A,A_0}	$\rho_{N_{0/A},SD_{N_E}}$	Δ_A	$\Delta_{A'}$	Δ_{A_i}
DWUG DE	818	659	.040	.20	.19	.78	.08	0.74	0.74	0.81
DWUG EN	816	665	.039	.26	.22	.54	.08	0.78	0.83	0.88
DiscoWUG	342	301	.058	.20	.00	.41	.00	0.68	0.74	0.92
All	609	475	.044	.22	.20	.41	.07	0.72	0.76	0.90

Table 4: From Left to Right: Mean of annotations per WUG. Median of annotations per WUG. Relative number of zero-annotations for all WUGs. Mean variance of annotations per word usage pair, where only edges with at least two non-zero annotations were considered. Median variance of annotations per word usage pair. Spearman rank comparing the number of annotations against the number of zero annotations per WUG. Spearman rank comparing the mean relative number of zero annotations of a word usages annotation against its mean annotations standard deviation. Sum where triangular inequality holds against sum of triangles in all WUGs. Sum where triangular inequality holds against sum of triangles in all WUGs, where only word usage pairs were considered having no annotation disagreement. Sum where triangular inequality holds against sum of triangles in all WUGs, where triangles where calculated per annotator.

Zero Annotations and Variance Zero annotations, from the DURel scale (Table 1), are undesired annotations, as they do not provide much information. We see that the number of zero annotations rises with the overall number of annotations for a WUG, which shows that most zero-annotations are a spurious effect of the annotation process itself (see Figure 6 middle and Spearman's ρ_{A,A_0} in Table 4) and make around 4.4% of all annotations.

Word Usages with high number of zero annotations and high annotation deviation may be an indicator for ambiguous usages, where the assignment to a sense is not clear (Erk et al., 2009). Hence we also examine the correlation between the mean relative number of zero-annotations against its mean annotations standard deviation (Figure 6 right and Table 4 $\rho_{N_{0/A},SD_{N_E}}$) per word usage in a WUG. We are not able to observe a correlation between the rise of zero annotations and its

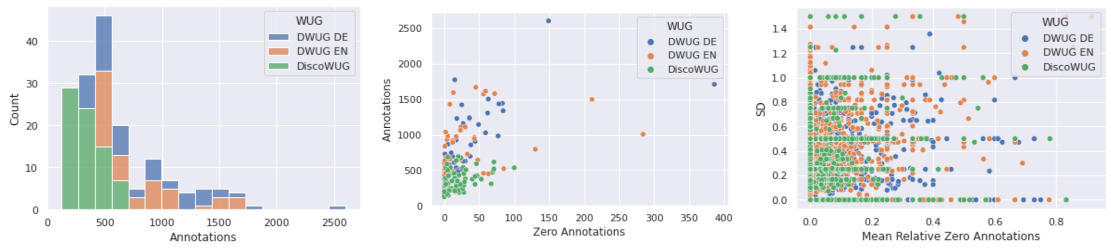

Figure 6: **Left:** Number of annotations per WUG. **Middle:** Number of zero-annotations against number of annotations per WUG. **Right:** A comparison per node, where the mean of relative zero-annotations of all edges is compared against the mean standard deviation of the annotations of all edges. We only consider edges of a node, if there are at least two non-zero annotations.

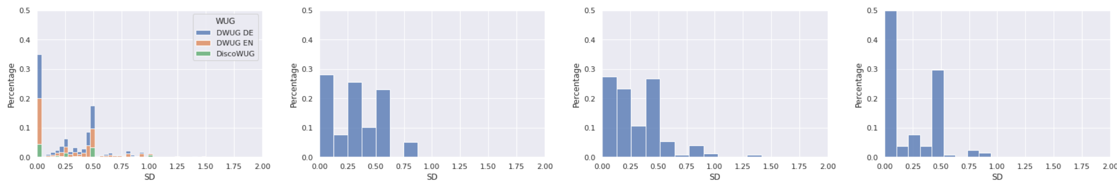

Figure 7: **Left:** Mean standard deviation of annotations for all word usage of all WUGs. **Middle Left:** Mean standard deviation of annotations for all word usages of *Zehner* from DiscoWUG. **Middle Right:** Mean standard deviation of annotations for all word usages of *abbauen* from DWUG DE. **Right:** Mean standard deviation of annotations for all word usages of *bag* from DWUG EN. We only included word usage pairs with at least two non-zero annotations.

standard deviation of a word usage, but do observe that there some nodes present in each data set that express high annotation disagreement and high number of zero-annotations. We suspect that there are some mechanisms that drives this behaviour. We do observe that the mean standard deviation per node does tend to be small, between zero and .5, suggesting that for most word usage pairs the semantic relatedness is well defined (Figure 7).

Figure 8 (left) shows that only a few edges are annotated more than once, which is due to how the word usage pairs were sampled (Schlechtweg et al., 2021b; Kurtyigit et al., 2021a). While Kurtyigit et al. (2021a) (DiscoWUG) only used a random sam-

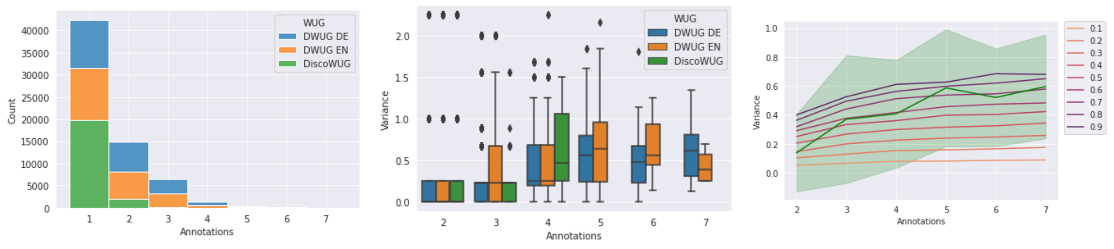

Figure 8: **Left:** Count of non-zero annotations per edge. **Middle:** Variance of annotation for different number of annotations. **Right:** Median variance of data set compared to simulated variance with a Poisson-Distribution and different λ parameters.

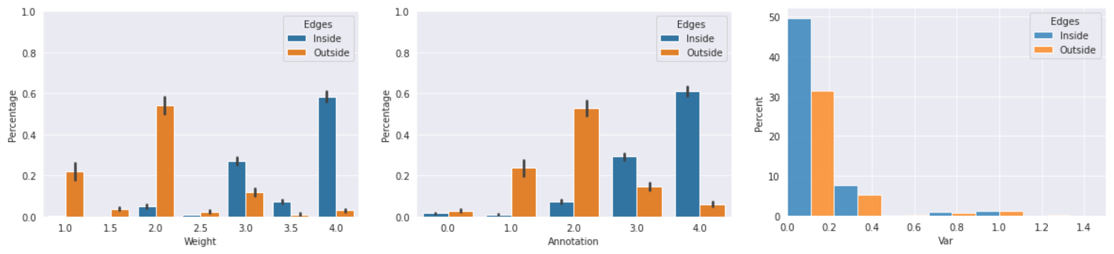

Figure 9: **Left:** Relative weight (median of annotations per word usage pair) distribution between senses for DWUG DE/EN and DiscoWUG. **Middle:** Relative annotation distribution between senses for DWUG DE/EN and DiscoWUG. **Right:** Variance of annotations per word usage pair between senses for DWUG DE/EN and DiscoWUG.

pling for word usage pairs, thus resulting in less multiple annotations, Schlechtweg et al. (2021b) re-sampled word usage pairs, where the annotation disagreement was not optimal. We simulate the effect of multiple annotators where the error is chosen based on a Poisson-distribution (Figure 8 right, excluding green curve) and also see the effect of rising variance, which indicates that this effect occurs naturally, but are not able to reproduce this rapid rise in variance. Our observations about the annotation disagreement is in line with that of Erk et al. (2009), which observed that by using a graded scale the disagreement between annotations was not insignificant.

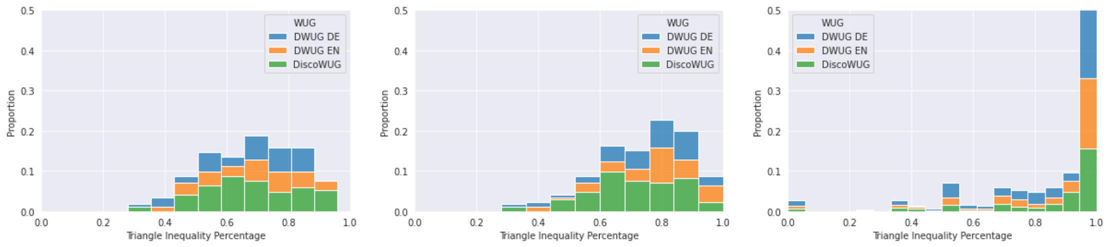

Figure 10: **Left:** Triangular inequality for DWUG DE/EN and DiscoWUG over all weights. **Middle:** Triangular inequality for DWUG DE/EN and DiscoWUG only considering edges with no annotation variance. **Right:** Triangular inequality for DWUG DE/EN and DiscoWUG where only edges are considered which are annotated by the same annotator.

Annotation Distribution We can observe that most annotations for word usages (Figure 9) in the same sense-cluster tend to have higher weighted edges, while the opposite effect is seen for word usages between two sense-clusters. This is what we would expect to see, as word usage pairs having the same sense should exhibit a high score on a graded scale (here DURel scale, Table 1), while word usage pairs assigned to different senses should exhibit lower scores. It is important to note that all WUGs were clustered using Correlation Clustering, that optimizes this behaviour. The distribution of weights and annotations seem to follow a binomial distribution.

What is interesting to mention is that we would expect the one-annotations to be more prevalent than the two-annotations for word usages pairs not belonging to the same sense. This indicates that annotators tend to annotate word usage pairs with different senses less likely with an extreme value. We observe that the whole scale was used, which is inline with Erk et al. (2009), as they also found that annotators tend to use the full scale.

Triangular Inequality The triangular inequality property in graded annotations is an important topic to consider. Erk et al. (2013) lists three important properties, that can be used to assist in the annotation process. We compared the triangular inequality property for the following three different cases (Table 4 and Figure 10). When all annotators are considered for the weight of the edge, the triangular property only holds in around 72%, which most likely is due to different annotators perceiving either the graded scale or word usage pair differently. When only considering word usage pairs where the annotator disagreement is zero, the triangular property only holds for around 76% of triangles. From Figure 8 (left), we observed that most edges are annotated only once, thus resulting in mixing different annotators annotations, which is not favourable. Considering every annotator separately, the triangular inequality holds in 90% of cases, which is close to the observation made by Erk et al. (2013). This is also a strong indicator, that annotators do not perceive a graded scale and the relatedness of word usage pairs in the same fashion, but do stay overall close to their own interpretation.

5 Simulation

For the simulation our goal is to reproduce the annotation procedure of WUGs, such that there is no need for human annotators or specific word usage graphs. The simulation can be described as follow. Before the annotation process starts, a **True WUG** is generated (Section 5.1). A *True WUG* $T = (V, E, W, C)$ is a weighted undirected complete graph, where vertices $v \in V_T$ represent a node, $v, u \in V_T : (v, u) \in E_T$ represent an edge between two nodes and $w_{v,u} \in W_T$ the weight of an edge. $C_T = \{C_i | i \in \mathbb{N}\}$ describes the clusters in a graph, where for each $C_i \in C_T : N \subseteq V_T : C_i = N$, for any $C_i, C_j \in C_T : C_i \cap C_j = \emptyset$ and $\cup_{i \in \mathbb{N}} C_i = V_T$. Thus, $\forall v \in V_T : \exists! C_i \in C_T : v \in C_i$. This *True WUG* can be conceptualized as a possible complete WUG where edge-weights and its corresponding clustering represents the correct representation of a WUG for an unspecified word.

Based on this *True WUG* a model performs its sampling (Section 5.3) and clustering (Section 5.4) steps. Each sampled edge is then annotated by some annotator, mirroring the behaviour of human annotators (Section 5.2), by introducing some annotation noise to the weight of the sampled edge. The whole process can be stopped, based on some stopping criterion (Section 5.5) defined by the model.

The resulting WUG is then an **Annotated WUG** $A = (V, E, W, C)$, which is an undirected weighted graph where $V_A \subseteq V_T$ and $E_A \subseteq E_T$. The weights W_A may differ to W_T, as well as the clustering of V_A compared to V_T. This *Annotated WUG* should then represent an annotated WUG with characteristics observed in Section 4, thus mirroring the real world annotation process of a word usage graph.

In the following subsections we will introduce some strategies used for the graph generation, annotation imitation and models. It is important to note, that no model may use information from the *True WUG*, which could not also be used during a real annotation process, meaning that any model is restricted to only being able to see what nodes are present in the *True WUG* and all information provided by the *Annotated WUG*. The annotation phase is an exception, as it is able to access the weight set of a *True WUG*, such that a realistic annotation is possible.

5.1 Graph generation

Graph generation is a critical part for the simulation, as generated *True WUGs* should exhibit similar characteristics as observed WUGs. The Stochastic Block Model (SBM) (Holland et al., 1983) is a simple generative model for random graphs and takes the following two parameters into account for its generative process.

- A number of nodes n, clustered into m disjoint sets $C_1, ..., C_m$
- A symmetric probability matrix $P \in \mathbb{R}^{m \times m}$, where each scalar value $P_{i,j}$ of P describes the probability of observing an edge between the nodes $v \in C_i$ and $u \in C_j$

The Weighted Stochastic Block Model (WSBM) (Aicher et al., 2014; Peixoto, 2017) is an extension to the SBM generative model by incorporating edge-weights for sampled edges. This is done by extending the generative model by the following parameter:

- A symmetric matrix D of size $m \times m$, which items consists of distribution $d_{i,j}$ from which the edge-weight of a sampled edge $v \in C_i$ and $u \in C_j$ is sampled

Schlechtweg et al. (2021a) showed, that by using this process it is possible to generate reasonable graphs which model WUGs, hence allowing us to generate undirected weighted graphs used in the simulation as the underlying *True WUG*.

5.2 Annotation

Section 4.3 shows that annotation noise, in form of error or zero annotations, is not an inconsiderable effect of the annotation process. Hence it is important to model this process, as the simulated annotation process should resemble the true annotation process as closely as possible. We model this effect by manipulating the weight $w_{v,u} \in W_T$ of a sampled edge from T and adding this manipulated weight $w'_{v,u}$ to A. In this sense, the *Annotated Graph* represents a noisy *True Graph*, thus mimicking the annotation process of WUGs. Only this process has access to the weights of the *True Graph*, as it is modelling the annotation phase.

Annotation Error The annotation error models the divergence of annotation from the 'true' annotation, as well as the divergence of annotators per edge. This is done in the following way. Given some edge $(v, u) \in E_T$ of T, sample an error e from a distribution D. This error is then randomly added or subtracted from $w_{v,u}$. The resulting weight will then be min/maxed against a predefined range. This range in most cases is defined by the extrema of the scale used to annotate the edges.

Zero Annotations This process models the indecisiveness of annotators, due to ambiguity or other reasons, by changing the sampled edge-weight to zero with a given probability p_0.

5.3 Sampling Algorithms

As seen in Section 4.1, most WUGs contain many nodes, resulting in an enormous expenditure for fully annotating such graphs. Considering the observations made in Section 4.3, the order of edges sampled and annotated plays an important part.

Thus sampling strategies should be employed in aiding to reduce the annotation complexity, by choosing edges in such a manner, such that a sufficient and efficient result can be derived quickly.

We model sampling as an algorithm that chooses two nodes $u, v \in V_T$ of the *True WUG* T and adding the node pairs as well as the corresponding edge to the *Annotated WUG* A by $V_A = \{u, v\} \cup V_A$ and $E_A = \{(u, v)\} \cup E_A$. The weight of this sampled edge is then modified by the two approaches described in Section 5.2 and added to W_A. The weight of an edge for observed WUGs is the median of all annotations for this edge. For simplicity we exclude the process of calculating the median and assume that the weight of an edge from an *Annotated WUG* is the median of all annotations added to this edge.

5.3.1 Random Sampling

Random sampling is the simplest sampling strategy, as it does not consider the current state of an *Annotated WUG* in any way. Given a *True Graph* T and an *Annotated Graph* A, for every step we sample randomly two nodes $u, v \in V_T$, where $v \neq u$, and add those pairs with their respective edge to the graph A. Thus any node-pair is equally likely of being sampled and it is possible that some pairs are sampled multiple times. We include this procedure, as it is an interesting base case scenario and was used by DiscoWUG to generate its word usage pairs.

5.3.2 Random Walk

Random Walk works in the following way. Given a *True WUG* T and an *Annotated WUG* A, randomly sample one node $v \in V_T$, which serves as our starting point. In every step sample a different node $u \in V_T$ and add the node-pair v, u and its corresponding edge to A. Node u now servers as the staring point for the next step. This leads to every node $v \in V_A$ at any step to be either directly connected or connected by a path to any other node in $V_A \setminus \{v\}$. Figure 11 illustrates the WUGs created by Random Walk.

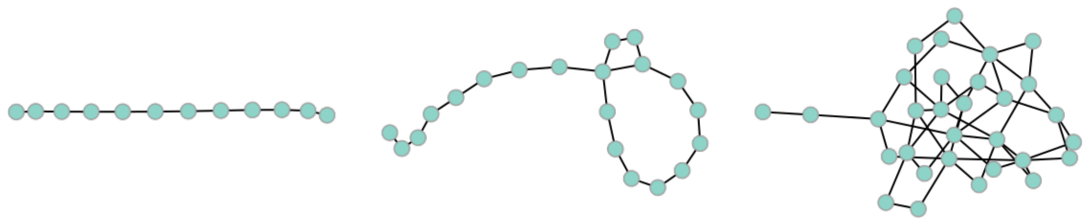
Figure 11: Illustration of a WUG produced by Random Walk at some steps and demonstrates that any added node is connected to any other node by some path.

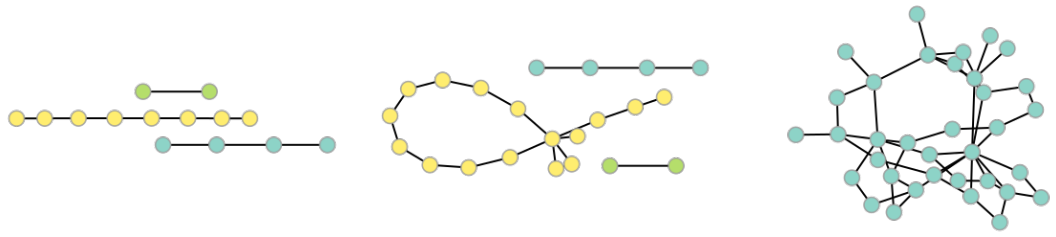
Figure 12: Illustrating the behaviour of PageRank and the resulting sampled WUG. The coloration of a node is for visualization of unconnected components produced by a jump.

5.3.3 PageRank

PageRank is one of the well known algorithms for ranking web pages, by assigning a score to each node in the web graph. We leverage the idea of PageRank, by using a Random Walk combined with a teleportation probability t. The sampling steps are mostly equivalent to the Random Walk algorithm as described in 5.3.2. The crucial difference is that in every step there is a probability, that the starting point node for the next step is replaced by a random node from V_T. This probability is driven by the teleportation probability t. Hence, in every step there is a probability t that the Random Walk jumps to a randomly chosen node and continues the Random Walk from there. This jumping behaviour is exemplified in Figure 12.

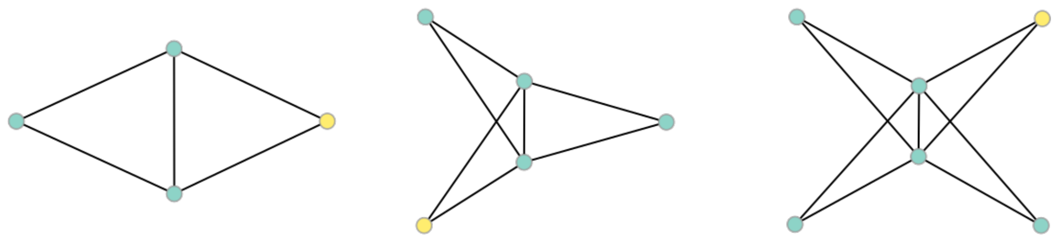

Figure 13: Example of MRW sampling steps, illustrating the prioritization of new nodes (yellow) as well as building denser structures.

5.3.4 Modified Random Walk

Modified Random Walk (MRW) builds on top of the Random Walk in the following way. Given a *True WUG* T and an *Annotated WUG* A, let F, S be two sets, where $F = \emptyset$ and $S = V_T$. Randomly sample a starting node v from S and update the set accordingly to $F = F \cup \{v\}$ and $S = S - \{v\}$. Now for every step the following is applied:

- Sample a node $u \in S$ if S is not empty. If S is empty, sample a node from F.
- Add the node-pair v, u and the corresponding edge to A
- Sample a node $n \in F$ and add the node-pair u, n and the corresponding edge to A
- Update the sets by $F = F \cup \{u\}$ and $S = S - \{u\}$
- Node n now servers as the starting point for the next step

The main idea is that we sample edges between two sets, a **Found Set** F and a **Search Set** S. The *Found Set* contains all visited nodes, while the *Search Set* contains only nodes, which currently are not in V_A, therefore nodes which are currently not in V_A are prioritized and any sampled node is initially connected to at least two nodes, that are already present in the *Annotated WUG*. The resulting graph is similar in its properties compared to Random Walk (see Section 5.3.2), but due to its modified walk prioritizes denser structures as well as adding new edges. This difference can also be seen in Figure 13.

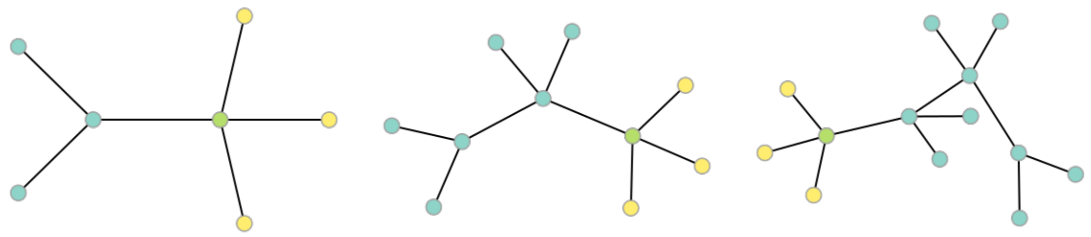
Figure 14: A possible traversal and sampling by MERW. Yellow nodes represent the sampled neighborhood for a given sampling step and the green node illustrates the node to which MERW traversed next.

5.3.5 Modified Search Space Random Walk

The Modified Search Space Random Walk (MERW) extends the Random Walk in the following way. Given a *True WUG T* and an *Annotated WUG A*, randomly sample one node $v \in V_T$, which serves as our starting point. Now in every step, k nodes are randomly sampled. For each sampled node a node-pair is constructed, containing the starting node and the sampled node, and added with their corresponding edge to A. After this, a node from this sampled set is chosen as the next starting node for the next step. This approach allows us to capture some information about a nodes neighbourhood. Figure 14 provides an example on how a WUG might be traversed and exemplifies the behaviour of MERW.

5.3.6 DWUG Sampling

This sampling strategy is a derivative of the sampling strategy employed by Schlechtweg et al. (2021b), and is the first approach that uses information gained from clustering.

Each step is characterised by the following two Phases. The **Exploration Phase** performs a Random Walk based on some set of nodes till m edges are created. The **Combination Phase** takes in a set of nodes and a set of clusters. For each node in the set chooses a node from each clusters and adds this edge to the *Annotated WUG*.

Given a *True WUG T* and an *Annotated WUG A*, the first step is defined in

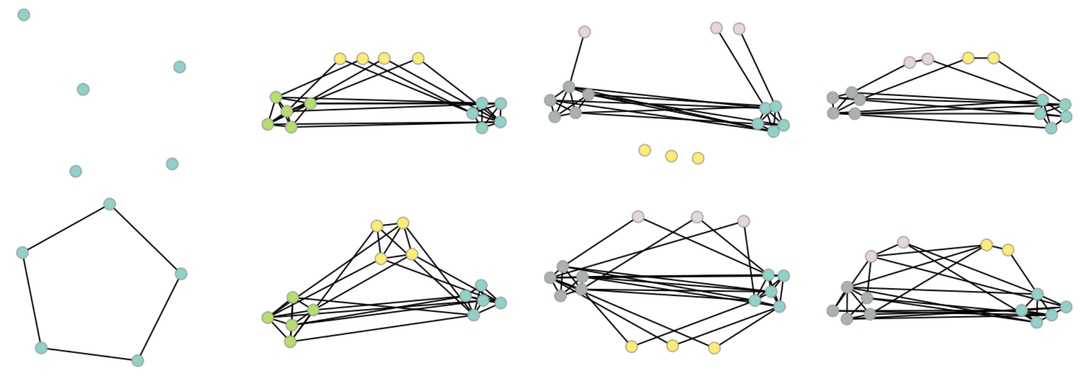
Figure 15: This is an illustration of the different phases of DWUG+RS Sampling strategy. **Left** is the initial seeding. **Middle left** shows the exploration phase performed on the yellow nodes, which do not belong to any cluster bigger than some threshold (green or blue). **Middle right** is an example of the combination phase performed on the purple nodes, which are currently not connected to all clusters bigger than some threshold (grey and blue) and the newly added nodes (yellow). **Right** highlights the intrinsic stopping criterion of DWUG and the random sampling thus performed by DWUG+RS.

the following way. Chosen randomly n number of nodes from V_T. On this set of nodes an *Exploration Phase* is performed. This step is only performed once and in a sense is a seeding step for the DWUG Sampling approach (Figure 15 left). After this initial step, each successive step can be considered as the normal loop for DWUG Sampling. Given that a clustering is known for A choose all nodes, where $n \in V_A \land n \in C_{A,i} : |C_{A,i}| = 1$. Hence, all nodes are chosen, where this node is the only member of its clustering assignment. Let this set be N_B. Now this set N_B is divided into two sets N_C and N_E with the following rule:

$$N_C = \{n | n \in N_B : \exists C_i \in C_A : |C_i| \geq s \land \forall u \in C_i : (n, u) \notin E_A\} \tag{2}$$

Thus, set N_C contains all nodes, which are the only member of its clustering class and is not connected to all clusters of at least size $s \in \mathbb{N}$.

$$N_E = N_B \setminus N_C \tag{3}$$

Set N_E contains all nodes, which are the only member of its clustering class, but are already connect to all clusters of at least size s. N_C is then extended by a set of n nodes which are randomly sampled from V_T, where $\forall v \in N_N : v \notin V_A$ holds. For this extended set N_C a *Combination Phase* is executed with all clusters larger than s (Figure 15 middle right). For set N_E an *Exploration Phase* is executed, with a given number of edges m that have to be created (Figure 15 middle left).

This sampling strategy splits nodes into two groups, first one being a set of nodes N_C for which there is unknown information of its relation to other clusters. The *Combination Phase* adds edges for those nodes, hence trying to find a cluster that this node may belong to. This set is also expanded by yet not added nodes, as their relation to other clusters is not known. N_E is a set of nodes, where its relation to other cluster is known, but do not appear to be a part of any of those bigger clusters. Thus an *Exploration Phase* is performed on these nodes to add relational information, that may lead to a new cluster.

5.3.7 DWUG+RS Sampling

DWUG Random Sampling (DWUG + RS) is an extension to the *DWUG Sampling* approach from Section 5.3.6. In any given step after the first step, it is possible that *DWUG Sampling* will not sample any new edges. This is due to if $\forall C_i \in C_A : |C_i| \geq s$ and $V_A = V_T$, meaning that if all clusters are bigger than some threshold and all nodes from V_T are added to V_A, no nodes are selected for any of both phases, thus no edges are sampled. This is an intrinsic stopping criterion of *DWUG Sampling*, which may not always be desired, as nodes may only be sparsely connected and may lead to bad results. Therefore we extended *DWUG Sampling* by adding a random sampling. If both sets N_C and N_E are empty, n random sampled edges are added to A (Figure 15 right). Random Sampling may be substituted by any other sampling strategy.

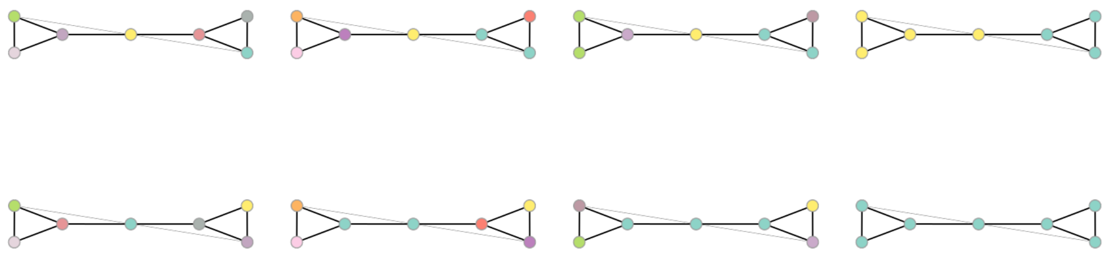

Figure 16: Illustrating two possible clustering (top and bottom) achieved by CW and illustrates how the initial ordering of nodes for the iteration step may impact the resulting clustering. We can also observe that the middle node (yellow) is trapped between two ideal clusters.

5.4 Clustering Algorithms

Clustering is the back bone of finding senses in an annotated WUG. There are many important things to consider in choosing a good clustering strategy. From observation made in Section 4.2 and Section 4.3 it is important, that a clustering strategy is able to find small clusters and is robust against noise. As we use clustering on the *Annotate Graph*, the performance heavily depends on the chosen sampling strategy from Section 5.3, as well as the noise introduced during the annotation process (see Section 5.2). Thus it is important that the clustering strategy is impervious against effects introduced by these models, like sparse degrees for nodes and high annotation error between and inside clusters.

5.4.1 Chinese Whispers

Chinese Whispers (CW) originates from the need for a low computational effort clustering strategy for graphs with large number of nodes and edges, and was developed for problems in Natural Language Processing (Biemann, 2006). The basic idea is that the assignment of a node to a cluster is based on the local neighbourhood of that nodes and works the following way. Given an *Annotated Graph A*, first

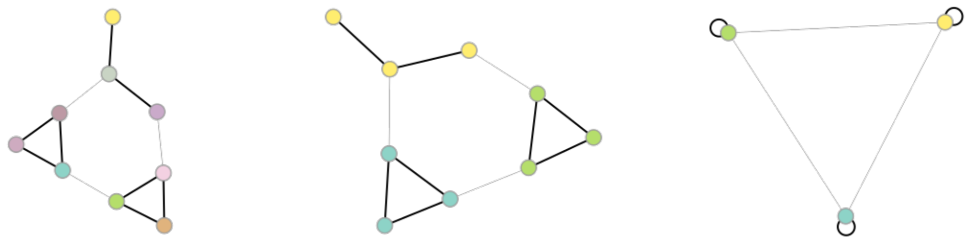

Figure 17: An example of the different steps in Louvains method. **Left** represents the initialization step, **middle** the cluster assignment step and **right** the combination step.

an **Initialization step** is performed where all nodes V_A are assigned to their own clusters. Next an **Iteration step** is performed, where for every node $v \in V_A$ the following rules are applied:

$$C_{new} = argmax_{C_i \in C}(\sum_{u \in C_i} w_{v,u}) \cup \{v\} \tag{4}$$

$$C_v = C_v \setminus \{v\} \tag{5}$$

This means that a node v is assigned instantaneously to the cluster where its sum of edge-weights is the highest. If there are multiple possible cluster assignments, a cluster is chosen by random. The *Iteration step* is repeated, until there are no changes in the clustering assignment or until some number of iterations are done. The order of how nodes are chosen is determined randomly. Biemann (2006) mentions, that there might be some nodes which are assigned to different clusters every iterations, as their sum of edge-weights to different clusters are the same. This behaviour may prove disadvantageous, as discussed in Section 4.2 some senses might only contain one word usage, thus may not be found by this approach. An example of the clustering strategies *Iteration step* can be found in Figure 16. Chinese Whispers provides a hard clustering with a time complexity of $\mathcal{O}(|E|)$.

5.4.2 Louvain Method

Louvain Method was introduced by Blondel et al. (2008), which optimizes *modularity* to find an optimal clustering for an given graph. *Modularity* is given by the following equation:

$$Q = \frac{1}{2m} \sum_{v,u \in A} (w_{v,u} - \frac{k_v k_u}{2m}) \delta(C_v, C_u) \tag{6}$$

$$m = \frac{1}{2} \sum_{w \in W_A} w \tag{7}$$

$$k_v = \sum_{u \in V_A} w_{v,u} \tag{8}$$

Where m is the sum of all weights in a graph, k_v is the sum of all weights connected to node v and δ is the Kronecker delta function, which if both nodes are in the same cluster, resolves to one. By optimizing this function, the edge density inside a cluster is maximized, while the edge density between two clusters is minimized. Louvains method does this in the following way. Given an *Annotated Graph A*, first an **Initialization step** is performed where all nodes V_A are assigned to their own clusters. Next a **Cluster Assignment step** is performed. In this step, each node v removed from its current cluster and is moved into each neighbouring cluster and the gain in *modularity* is evaluated. If a cluster exists, for which the gain is positive and the maximum, node v is moved into this cluster. If none of these gains is positive, the node stays in its current cluster. The gain of *modularity* by moving an node v into a cluster C_i is is calculated using the following rule:

$$\Delta Q = \left[\frac{\sum_{in} + k_{v,in}}{2m} - \left(\frac{\sum_{tot} + k_v}{2m} \right)^2 \right] - \left[\frac{\sum_{in}}{2m} - \left(\frac{\sum_{tot}}{2m} \right)^2 - \left(\frac{k_v}{2m} \right)^2 \right] \tag{9}$$

$$\sum_{in} = \sum_{v,u \in C_i} w_{v,u} \tag{10}$$

$$\sum_{tot} = \sum_{v \in C_i, u \in V_A \setminus C_i} w_{v,u} \tag{11}$$

$$k_{v,in} = \sum_{u \in C_i} w_{v,u} \tag{12}$$

$$k_v = \sum_{u \in V_A} w_{v,u} \tag{13}$$

$$m = \frac{1}{2} \sum_{w \in W_A} w \tag{14}$$

(15)

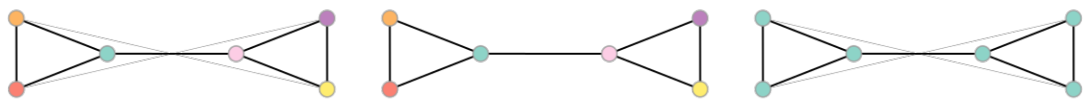

Figure 18: Example of the CCC, showing the individual steps performed. **Left:** Initial WUG. **Middle:** Edge removal step. **Right:** Connected component search.

$\sum_{in}$ is the sum of all edge-weights in cluster C_i, $\sum_{tot}$ the sum of all edge-weights for edges going into the cluster C_i, $k_{v,in}$ the sum of all edge-weights for edges going from node v to nodes in cluster C_i and k_v as well as m are defined as above. The change in *modularity* by removing node v from its current cluster is calculated in a similarly.

This step is repeated for all nodes, as long as there is change in the cluster assignment. The order of nodes has to be respected during each *Cluster Assignment step*. If a stable clustering is found, a **Combination step** is performed. This steps combines all nodes in a cluster and edges between clusters are combined, where the sum of the edge-weights between two clusters is the new edge-weight. Both steps, *Cluster Assignment* and *Combination step* are repeated, till no change in the cluster assignment occurs. An example of each step can be found in Figure 17. This clustering provides a hard clustering with time complexity of $\mathcal{O}(|V| \cdot \log |V|)$.

5.4.3 Connected Component Clustering

Connected Component Clustering (Hopcroft and Tarjan, 1973) exploits the idea of connected components, where only nodes which are either directly connected or connected by a path should be in the same cluster. We extend this by only considering edges, which are above some given threshold t.

Given an *Annotated WUG A*, first an **Edge Removal** is performed, where all edges from A are removed for which $w_{v,u} < t$ holds. Next a **Connected Component Search** is performed on the modified *Annotated WUG*, for which a node is selected

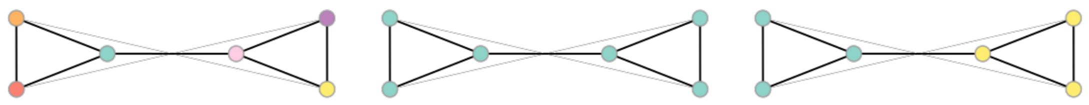

Figure 19: Example of how DWUG CC (**right**) finds a better clustering for a given WUG (**left**) compared to CCC (**middle**).

and a breadth-first or depth-first search is performed and any visited nodes are marked. If no more nodes can be reached by any of the marked nodes, a connected component is found. This process is repeated until all nodes of the modified WUG are marked. The connected components found during the search form the clustering for the given WUG. An example of this clustering strategy can be found in Figure 18.

This process can be optimized by modifying the breadth-first or depth-first search to only traverse edges above the given threshold t, thus the time-complexity of this clustering is given by the search function, which in both cases are $\mathcal{O}(|V| + |E|)$ and returns a hard clustering for the given WUG.

5.4.4 DWUG corelation clustering

DWUG Correlation Clustering (DWUG CC) (Bansal et al., 2004; Schlechtweg et al., 2020; 2021b) is a more specific clustering strategy geared towards WUGs, as it exploits the meaning of edge-weights. As discussed in Section 3, word usage pairs are annotated by a weight corresponding to its semantic relatedness (see Table 1). By considering this edge-weight information, DWUG CC minimizes the following:

$$(16) \qquad L(C_A) = \sum_{v,u \in \Phi_{E_A,C_A}} w_{v,u} + \sum_{v,u \in \Psi_{E_A,C_A}} |w_{v,u}|$$

$$(17) \qquad \Phi_{E_A,C_A} = \{(u,v) | u,v \in V_A : w_{v,u} \in W_A : C_{A,v} \neq C_{A,u} \wedge w_{v,u} \geq 2.5\}$$

$$(18) \qquad \Psi_{E_A,C_A} = \{(u,v) | u,v \in V_A : w_{v,u} \in W_A : C_{A,v} = C_{A,u} \wedge w_{v,u} \leq 2.5\}$$

Thus, this clustering minimizes low annotation scores inside a cluster and high annotation score between clusters, or worded differently, it maximizes the semantic relatedness of word usages inside a cluster while maximizing the dissimilarity of word usages between clusters. This approach may also cluster nodes together, which are not connected by edges with weight ≥ 2.5, which might be splitted into separate clusters. Figure 19 shows an example how DWUG CC finds a better clustering compared to CCC, due to the function being optimized. The DWUG CC provides a hard clustering for the given WUG.

5.4.5 Weighted Stochastic Block Model

The WSBM approach can not only be used as a generative process for graphs, but can also be used to infer a clustering for a given graph (Peixoto, 2014a; Schlechtweg et al., 2021a). The clustering approach consists roughly of the following two parts.

In the **Agglomeration** step a merge of the current clustering is performed, where each node is moved into a cluster based on some probability and ranked on how well they minimize a certain metric. Based on this ranking of moves, clusters are merged together. This is done till a desired number of clusters is reached. The **Markov Chain Monte Carlo** step is performed between each *Agglomeration* step. Here each nodes clustering assignment is modified by some probability, thus allowing some nodes to move between clusters. For further discussion regarding inferring clusters refer to Peixoto (2014a) and Schlechtweg et al. (2021a). The time-complexity for *WSBM* is $\mathcal{O}(|E| \log^2 |N|)$ and provides a hard clustering for the given WUG.

5.5 Stopping Criterion

For the model to be efficient and effective, we need a way to determine when to stop an annotation process. These have to capture a point, where a reasonable state in the annotation process is reached meaning that enough information is gathered, such that the clustering reflects the sense structure. If the annotation process is terminated to early, the resulting clustering might not reflect or capture the underlying sense structure completely or in the worst case be wrong. If too long, a reasonable solution might have already be found and by continuing the process only marginal information is added, while the annotation labor increases. Finding this delicate balance is the main essence of a stopping criterion.

5.5.1 Bootstrapped JSD

Bootstrapping (Efron, 1979) is the method by which a distribution is randomly sampled and a metric is calculated, such that some observation can be made about the underlying distribution. The Confidence Interval (**CI**) describes an interval, in which we expect an observed value to fall between, with some probability. The Jensen–Shannon divergence (**JSD**) (Endres and Schindelin, 2003) is a measurement of similarity of two distributions defined in the following way:

$$JSD(P||Q) = \frac{D(P||M) + D(Q||M)}{2} \tag{19}$$

$$M = \frac{P + Q}{2} \tag{20}$$

Where P and Q are two distributions and $D(P||Q)$ is the Kullback–Leibler divergence. The bounds of JSD are $[0, 1]$, where 0 means that distributions P and Q are the same. We combine these three approaches in the following way.

Let A be an *Annotated WUG*, we random sample with replacement the cluster distribution C_A n times. Let $\overrightarrow{\mathrm{C_A}}$ be the vector representation of the clustering, where each scalar represents the number of nodes in the cluster and is ordered by size. On both vector representations $\overrightarrow{\mathrm{C_A}}$ and $\overrightarrow{\mathrm{C'_A}}$, where C'_A is the sampled, we calculate $JSD(\overrightarrow{\mathrm{C_A}}||\overrightarrow{\mathrm{C'_A}})$. This is done for several rounds, after which we have a sample of JSD metrics with size m. On this sample size we calculate a higher percentile P and if its value is below some threshold t, we accept and stop the annotation process.

5.5.2 Gambette

The Gambette method Gambette and Guénoche (2011) is an approach to qualitatively analyse the robustness of clustering. We use Gambettes approach as follows.

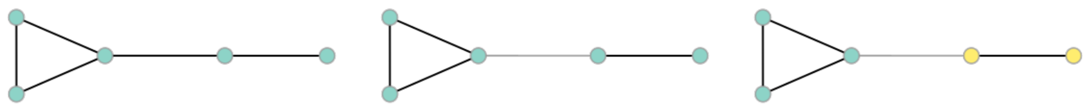

Figure 20: An example round of Gambette, where **left** is the initial WUG, **middel** represents the perturbed WUG by the random annotator and **right** is the resulting new clustering for this modified WUG. Based on the left and right WUG the ARI score is calculated.

Let A be an *Annotated WUG*. We make an identical copy A' of A and on this copy we let a random annotator annotate a percentage p of edges. The behaviour of the random annotator is driven by randomly choosing how to annotate an edge, where only edges are considered, which are already annotated at least once. This modified graph A' is then clustered and the adjusted Rand index Rand (1971) is calculated between A and A' as the robustness score of A. This is done for several rounds, after which we have a sample of robustness scores for A. Based on this sample we calculate the mean and if its value is higher than some threshold t we accept and stop the annotation process.

6 Experiment Setup

A model, as discussed in Section 5, consists of annotators, a sampling and clustering strategy, as well as an stopping criterion. We choose our models as all possible combinations of the previously mentioned sampling, clustering and stopping strategies. This is due changing any part of the model may lead to completely different results. For the annotation process we chose to model both the annotation error and zero annotations. In the following Sections we will discuss how we generated the *True WUGs* and the parameters used for the different models. Due to keeping the models as similar to each other as possible, we did not change the parameters based on other models attributes (i.e. clustering, sampling).

Graph Generation For the graph generation process we make use of the generative model WSBM mentioned in Section 5.1 and use the implementation provided by Chung et al. (2019).[2] As discussed in Section 3, we utilize two two sets of WUGs, *Coarse WUGs* and *Fitted WUGs*, to evaluate the models. For *Coarse WUGs* we use the observation made in Section 4 and selected the parameters for WSBM in the following way:

- Number of n nodes to be 100
- $m \in \{1, 3, 7, 10, 20\}$
- Log-normal distribution to dispense the nodes across the m clusters
- P as a unit matrix, where all entries are 1, to generate complete WUGs
- D, where each $d_{i,j}$ is a binomial distribution, with parameters $n = 3$ and $p = 0.99$ where $i = j$ and $1 - p$ for cases where $i \neq j$

The distribution matrix D was chosen this way, as we observe that the edge-weight distribution follows a binomial distribution most closely, where edges for nodes inside clusters are strongly annotated while edges between clusters are annotated with a low annotation score (compare Section 4.3 and Figure 9). For *Fitted WUGs* we used the data sets DWUG DE/EN as well as DiscoWUG to generate the *True WUGs*. As each WUG already contains information about its clustering of nodes and their associated edge weight, we use WSBM to statistically infer complete WUGs from each WUG in the data set. Due to the limitation of Chung et al. (2019) implementation, we use the approach described by Schlechtweg et al. (2021a) and the implementation provided by Peixoto (2014b) to infer the distribution matrix D and its parameters.

[2]`https://github.com/microsoft/graspologic`

Annotation Each experiment was run once using one and five annotators per sampled edge. Each annotator used shares the same parameter, based on the observations made in Section 4.3. For the *Annotation Error* we chose a Poisson-distribution with $\lambda = .35$, we think this models the annotation error as described in Section 4.3 most closely, if we take into consideration that edges in DWUG DE/EN were resampled based on high annotation disagreement. Hence choosing the variance more closely related with two annotators, being a Poisson-distribution with $\lambda = .3$. The *Zero Annotation* is sampled with a probability of 3.3%, which is lower by 1.1% from the observation made in Section 4.3. Each annotator was randomly chosen to annotate a sampled edge and as some sampling strategies allow an edge to be sampled multiple times (for example random sampling), it is possible that some edges are annotated multiple times.

Sampling As *Random Sampling*, *Random Walk* and *MRW* are autonomous sampling strategies, there is no need to specify any parameters. For *PageRank* we choose the teleportation probability t to be 0.1 and for *MERW* we chose the sample size k of the neighborhood as 2. For the *DWUG Sampling* and *DWUG+RS Sampling* parameters we chose the added nodes n and found edges m to be ten. As *DWUG+RS Sampling* also has the additional random sampling parameter n, we chose this to be ten.

Clustering For all clustering strategies we use a modified *Annotated WUG*, where all edge-weights were subtracted by 2.5, except *DWUG CC*, as it intrinsically subtracts the weights by 2.5. We argue, that the relatedness between two nodes in a WUG can be better captured by using this shifted scale for clustering. As *Louvain Method* does not work with negative edge-weights, we did not modify the edge-weights for this clustering strategy. For the *WSBM* we used the same approach as Schlechtweg et al. (2021a), as it shows promising results. We kept the model selection static as a binomial distribution, based on the observation made in Section 4.3 and by Schlechtweg et al. (2021a).[3] We also included both modes for *DWUG CC*,

[3]Implementation used from Peixoto (2014b) at `https://graph-tool.skewed.de/`

where nodes which are not connected by edge-weights ≥ 2.5 were split into separate clusters and kept together.[4]

Stopping Criterion For the *Bootstrap JSD* stopping criterion we used a sample size of 30, where for each sampled distribution 100 samples were drawn from the *Annotated WUG* and the percentile P was chosen as .975. For *Gambette* we choose a sample size of 10, the percentage p of randomly annotated edges to be 10% and the clustering to be the same as the models clustering. We did not specify a threshold value t for both stopping criterions, as we are interested in how these criterions develop over the number of annotations. This is why we also included them in the data collection process.

Data Collection As we were interested how the models develop over a rising number of annotations, we collected the adjusted Rand index, Jensen–Shannon divergence and the results from the stopping criterions for each model at 10, 20, 30, 40, 50, 100, 200, 300, 400, 500, 1000, 2000, 3000, 4000 and 5000 annotations. We believe that these points cover the most interesting annotation points, as low number of annotations (10-50 annotations) may show that some models perform good at low number annotations, mid-points (100-500 annotations) are a good measurement point for the models general performance, as observed WUGs have around the same number of annotations and late-points (1000-5000 annotations) may provide more information about a specific model drawbacks and advantages.[5]

[4]For CW we used the implementation provided by Ustalov et al. (2019) which can be found at `https://github.com/nlpub/chinese-whispers-python`. Louvain Method can be found at `https://github.com/taynaud/python-louvain`. WSBM was provided by Peixoto (2014b) at `https://graph-tool.skewed.de/`. DWUG CC and CCC was provided both by Schlechtweg et al. (2021b) at `https://github.com/Garrafao/WUGs`

[5]The whole simulation framework can be found at `https://github.com/confusedSerge/wug_sampling`

7 Analysis

The following sections will be divided into three parts. First we will cover how well our generated graphs capture the observed characteristics of DWUG DE/EN and DiscoWUG (Section 7.1). After that we will analyze the results of our models based on *Coarse WUGs* (Section 7.2). Finally we will compare our models against the *Fitted WUGs* (Section 7.3).

For comparatively measuring the performance of models and approaches we used the adjusted Rand index (**ARI**) and the inverse Jensen–Shannon divergence (**iJSD**) where *Annotated WUGs* are compared against their respective *True WUGs*. *ARI* is evaluated on the cluster assignment of nodes present in *Annotated WUGs*, depicting how well the true clustering has been recovered. For *iJSD* we use the sorted vector representation from Section 5.5.1, calculating the *JSD* between both WUGs and form the inverse. Hence this metric represents how accurate the cluster distribution has been captured by the model. The main drawback by using this approach is that for particular big WUGs with a heavy Log-distribution of nodes per cluster the metric lacks in capturing small deviations, in particular when small clusters are merged.

For example, let $\overrightarrow{C_T}$ of a *True WUG* be $(90, 6, 3, 1)^T$ and $\overrightarrow{C_A}$ of an *Annotated WUG* be $(100, 0, 0, 0)^T$. The *iJSD* would give a score of 0.964, while *ARI* would give a score of 0. We still include this metric, as it does provide insight into the distribution of nodes to clusters. Considering the case where $\overrightarrow{C_T}$ is $(100, 0, 0, 0)^T$ and $\overrightarrow{C_A}$ is uniformly distributed, we get a distance score of 0.619. Thus *iJSD* can be leveraged as a coarse indicator for the cluster distribution.

7.1 Generated WUGs comparison

Analysing characteristics of the generated WUGs is important, as those may be the main factor on how well a model or its components perform. Arguably the most important point is how well the *Coarse WUGs* and *Fitted WUGs* compare to and

	$\lvert\bar{C}\rvert$			$\lvert\tilde{C}\rvert$			$\lvert\tilde{C}_0\rvert$			$\lvert\tilde{C}_1\rvert$			$\lvert\tilde{C}_2\rvert$			$\lvert\tilde{C}_{i\geq3}\rvert$		
	Observed	Coarse	Fitted	Observed	Coarse	Fitted	Observed	Coarse	Fitted	Observed	Coarse	Fitted	Observed	Coarse	Fitted	Observed	Coarse	Fitted
DWUG DE	5.7	3.6	2.8	4.5	2.0	2.0	.84	.77	.76	.09	.14	.15	.01	.00	.00	.00	.00	.00
DWUG EN	8.2		2.8	5.0		2.0	.92		.84	.05		.09	.02		.00	.00		.00
DiscoWUG	4.8	12.6	3.8	4.0	11	3.0	.77	.81	.80	.12	.02	.12	.02	.01	.02	.00	.00	.00
All	5.9	8.1	3.2	5.0	5.0	2.0	.85	.80	.82	.08	.05	.12	.02	.01	.00	.00	.00	.00

Table 5: Mean and median of clusters and their relative sizes.

capture the observed WUGs, giving credibility to the models performance and its use case in non-simulated annotation processes.

For comparing the generated WUGs against observed WUGs, we choose the models most closely resembling the annotation process of DWUG DE/EN and DiscoWUG. In the case of DiscoWUG this would be the model consisting of one randomly chosen annotator per edge, where edges are randomly sampled and *DWUG CC* as the clustering approach (Kurtyigit et al., 2021b). DWUG DE/EN model is similar to the DiscoWUG model, but uses the sampling approach introduced by Schlechtweg et al. (2021b). The *DWUG Sampling* methodology does not implement re-sampling edges with disagreement or sampling between multi-clusters, which is why we choose to increase the annotations per sampled edge up to 5 to counteract this discrepancy. We selected *Annotated WUGs* with 500 annotations for comparison, as we have seen in Section 4 that most observed WUGs tend to have around this many annotations. In the following Sections we will refer to DWUG and DiscoWUG as the models used to generate the observed, coarse and *Fitted WUGs*.

Number of Senses From Figure 21 and Table 5 we can observe, that there is some discrepancy in the number of clusters found by the models compared to the observed data. For the *Coarse WUGs* this is mostly due to that an equal amount of underlying *True WUGs* with different amount of clusters were chosen for the simulation, explaining the more uniformly distributed cluster number. The fitting process of the observed WUGs for the *Fitted WUGs* tends to generate less clusters (Schlechtweg et al., 2021a), in most cases around 1 to 5 clusters, resulting in less clusters.

The skewness of the DWUG model for the *Coarse WUGs* can be explained by the

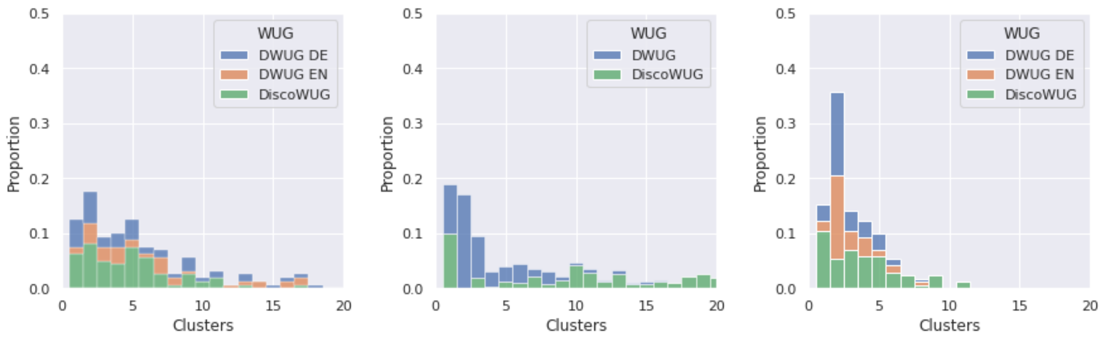

Figure 21: **Left:** Number of clusters (senses) in observed WUGs. **Middle:** Number of clusters for *Coarse WUGs*. **Right:** Number of clusters for *Fitted WUGs*.

number of annotators used. As we use 5 annotators per sampled edge, the number of possible sampled edges falls drastically and in addition with DWUG sampling, where it is possible that multiple edges are sampled per node, the probability of sampling an edge to a node belonging to a small cluster, as the clusters are log-distributed, is very low. Combining this with WUGs containing many clusters, it is even more improbable discovering these small clusters.

We do observe that the models reproduce the underlying *True WUGs* cluster number rather well, hence if the cluster number for the *True WUGs* were similar to the observed WUGs for the simulation, we would expect a similar distribution to that of the observed WUGs.

Sense Size Figure 22 shows that we are able to reproduce the Log-distribution of cluster sizes. This is expected, as we are able to control the distribution of nodes per cluster, yet we do observe that the *Fitted WUGs* follow the observed WUGs distribution more closely (see Table 5), which is even more visible comparing the distribution for each WUG independently (Figure 23), hinting at that the general distribution seems to follow a Poisson-Distribution, as discussed in Section 4.2.

Zero Annotations and Variance The percentage of zero annotations for coarse and *Fitted WUGs* is around 3.3% (Table 6). This is lower than the percentage on

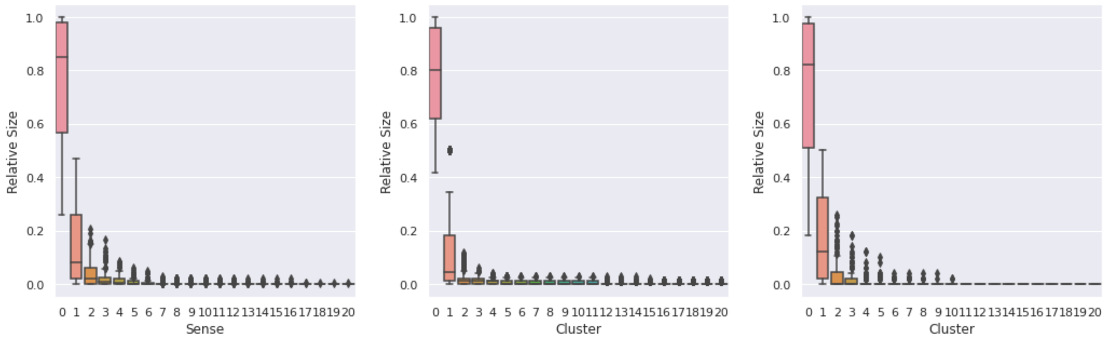

Figure 22: **Left:** Relative sense size for all WUGs of the data set. **Middle:** Relative cluster (sense) size for all *Coarse WUGs*. **Right:** Relative cluster (sense) for all *Fitted WUGs*.

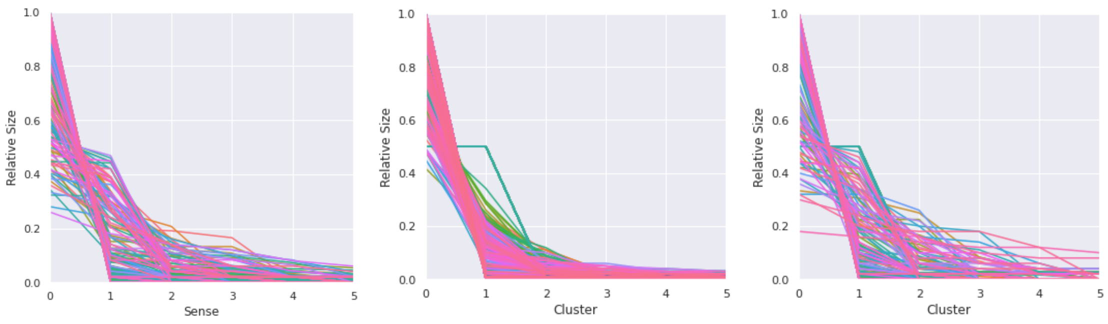

Figure 23: **Left:** Relative sense distribution for single WUGs for observed WUGs. **Middle:** Relative cluster (sense) distribution for single WUGs for *Coarse WUGs*. **Right:** Relative cluster (sense) distribution for single WUGs for *Fitted WUGs*.

the observed WUGs, which is expected, as we choose a zero annotation probability of .033 for the simulation. This shows that it is possible to reproduce the amount of zero annotations during the simulation and we expect that if the probability for the simulation was chosen to be .044 we would observe this.

We were also able to reproduce similar Spearman rank scores for $\rho_{N_{0/A}, SD_{N_E}}$ (Figure 24 and Table 6), indicating that for most nodes the number of zero annotations does not correlate to its annotation error. However, we are not able to observe the same frequency of nodes with high standard deviation and zero annotations (Figure 24), consolidating the concept that there is some relation between high annotation

	$\frac{\lvert A_0\rvert}{\lvert A\rvert}$			$\bar{\mathrm{Var}} A_E$			$\tilde{\mathrm{Var}} A_E$			$\rho_{N_{0/A}, SD_{N_E}}$			Δ_A		
	Observed	Coarse	Fitted	Observed	Coarse	Fitted	Observed	Coarse	Fitted	Observed	Coarse	Fitted	Observed	Coarse	Fitted
DWUG DE	.040	.032	.031	.20	.17	.21	.19	.16	.16	.08	.00	.00	.74	.94	.73
DWUG EN	.039		.035	.26		.22	.22		.16	.08		.03	.78		.68
DiscoWUG	.058	.033	.032	.20	.11	.15	.00	.00	.00	.00	-.01	-.03	.68	.68	.51
All	.044	.033	.033	.22	.15	.19	.20	.00	.16	.07	.14	.12	.72	.69	.52

Table 6: Different metrics divided between the chosen models and the underlying WUGs used. From left to right: Proportion of zero annotation. Mean Variance of Annotation per Edge. Median Variance of Annotation per Edge. Spearman Rank of mean relative number of zero annotations of a word usages annotation against its mean annotations standard deviation. Proportion of where Triangular Inequality holds.

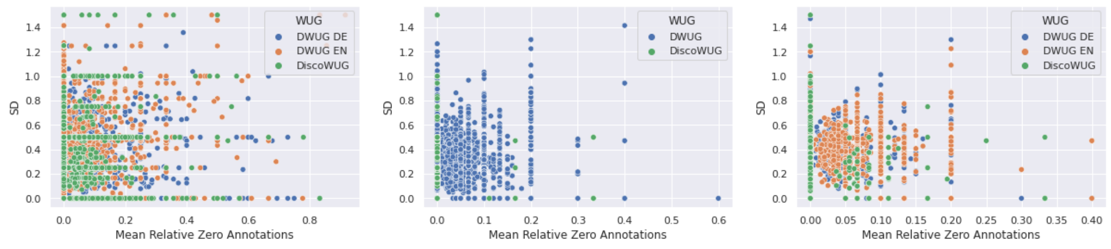

Figure 24: Mean Relative Zero Annotation against the Mean Standard Deviation per Node of the observed data (**Left**) compared against *Coarse WUGs* (**Middle**) and *Fitted WUGs* (**Right**).

disagreement and high number of zero-annotations, as discussed in Section 4.3.

The simulation produces similar distribution of mean standard deviation per node (Figure 25). We do observe that the standard deviation of 0 and .5 for the DWUG models is lower than the observed, which is a consequent of the chosen model, as each edge is annotated 5 times, thus more likely to produce a different standard deviation. The discrepancy for the DiscoWUG model stems from the number of nodes and annotations used. As all *Coarse WUGs* have 100 nodes, doubling that of the observed WUGs (Table 2), the probability of sampling an edge twice is lower. Figure 25 also showcases, that we are not able to reproduce nodes with high annotation error as frequent as for the observed WUGs.

In the case of annotation variance per edge, we are only moderately able to repro-

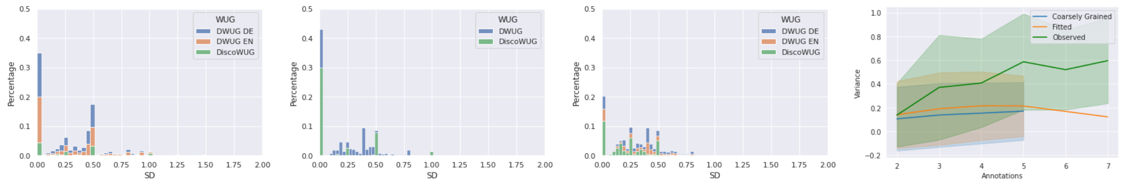

Figure 25: Mean standard deviation of annotations for all word usages of observed WUGs (**Left**), for all nodes (word usages) of *Coarse WUGs* (**Middle Left**) and for all nodes (word usages) of *Fitted WUGs* (**Middle Right**). **Right:** Comparison of median variance of edges per number of annotations of edges.

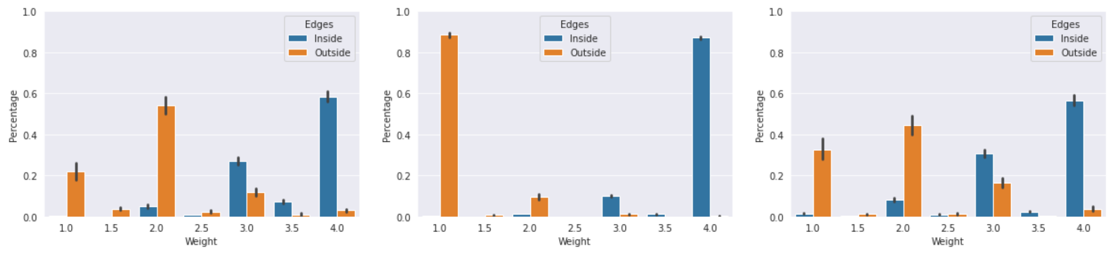

Figure 26: Relative weight distribution between clusters (senses) for observed WUGs (**Left**), for *Coarse WUGs* (**Middle**) and for *Fitted WUGs* (**Right**)

duce the same variance (Table 6), especially the observed effect of rising annotation disagreement between more than 2 annotators (Figure 25 right). The discrepancy is probably due to how edges were re-sampled in DWUG DE and DWUG EN, as edges with high annotation disagreement were annotated multiple times, thus leading to a higher variance.

Annotations We do reasonably well approximate the distribution of weights, annotations and variance of annotations for *Fitted WUGs* compared to the observed distribution (Figure 26, 27, 28). For the *Coarse WUGs*, this is not the case. Their weight and annotation distributions are more skewed towards their extremum. This is actually due to how we choose the parameter p for the distribution matrix D during the generation phase. As we choose the parameter p to be equal to .99 and .01 for the binomial distribution inside and between clusters respectively, we expected to observe the extreme values more often. From this we can assume that, if the

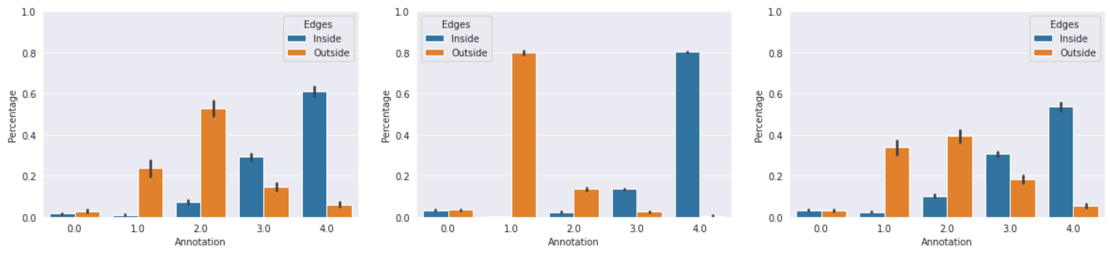

Figure 27: Relative annotation distribution between clusters (senses) for observed WUGs (**Left**), for *Coarse WUGs* (**Middle**) and for *Fitted WUGs* (**Right**)

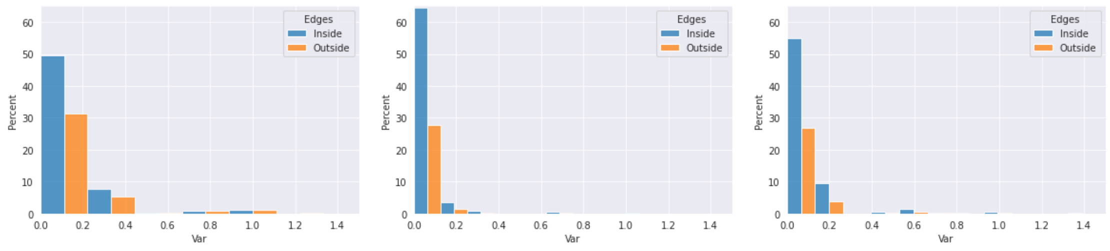

Figure 28: Variance of annotations per node-pair between clusters (senses) for observed WUGs (**Left**), for *Coarse WUGs* (**Middle**) and for *Fitted WUGs* (**Right**)

parameters are chosen carefully for the distribution matrix, for instance in the case of *Fitted WUGs*, where the parameters were inferred from the observed distribution, we are able to generate similar weight and annotation distributions.

Triangular Property We are only partially able to reproduce the triangular inequality. The DWUG model seems to be beneficial for the triangular inequality property, as opposed to the DiscoWUG model (Table 6). The main difference between these models are the number of annotator per edge and the sampling strategy employed. We also calculated the triangular property for edges which do not have an annotation disagreement, but did not observe a significant increase, which is in line with the observed data (Section 4.3). This leads us to believe that the sampling strategy used may be a big factor for the triangular inequality. If we compare the *Coarse WUGs* against the *Fitted WUGs*, we observe that the triangular property drops significantly. The main difference between these generated WUGs is their

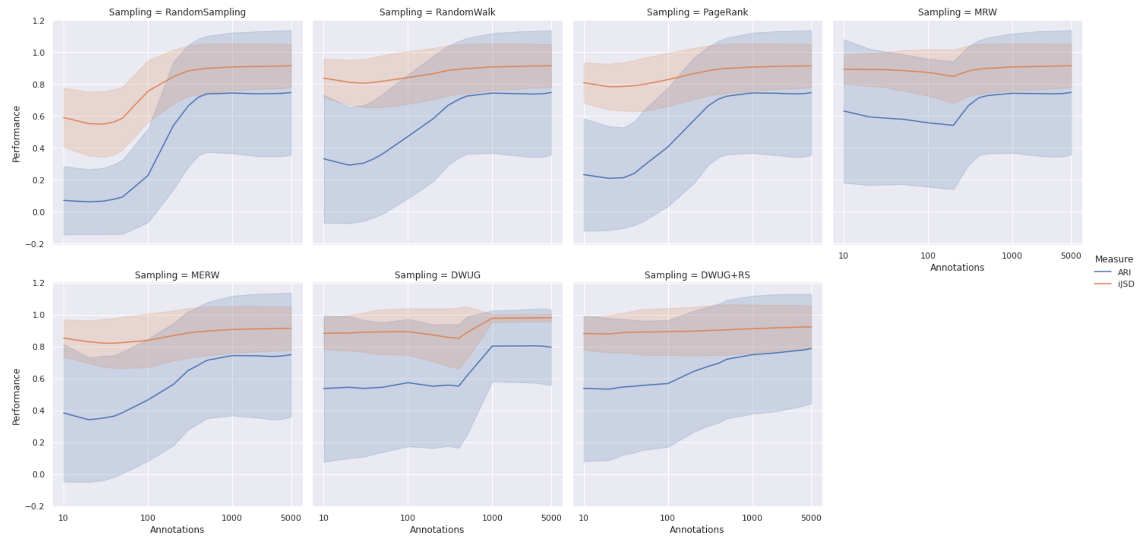

Figure 29: Overview of sampling strategies with one annotation per sampled edge. The data was generated by aggregating over all models.

weight distribution, where for the *Coarse WUGs* the distribution is skewed towards both extremum (Figure 26 and 27). This shows that the weight distribution is a significant factor for the triangular inequality.

7.2 Models on Coarse WUGs

In the following section we will compare the performance of models on *Coarse WUGs*. We will first take a look at the performance of each sampling strategy (Section 7.2.1), after which we will compare the clustering approaches (Section 7.2.2), then take a look at how the introduced stopping criterion's manage (Section 7.2.3) and finally on the overall performance of each model (Section 7.2.4).

7.2.1 Sampling

Random Sampling's performance between 10 and 100 judgments is not surprising, as it creates a lot of WU-Annotation-Pairs which are not connected (Section 7.2.1). This property of disconnectedness for small annotation sizes is being mitigated by doing a Random Walk across the WUG, thus only creating connected WU-Pairs, which improves the performance. While this approach does solve the problem of Random Sampling, it does bear the problem of cluster determination (Section 7.2.1). As PageRank combines both properties of Random Sampling and Random Walk, it is not surprising that this sampling strategy performs worse than than Random Walk. It incorporates the bad behavior of creating unconnected components, while performing better than Random Sampling, since it does not create these with the same frequency (Section 7.2.1). Modified Random Walk builds on top of the idea of keeping the graph connected, while extending this property by taking a path between the found and unknown set of nodes, thus reducing the undesired effects of Random Walk (Section 7.2.1) Modified Search Space Random Walk tries to improve Random Walk by exploring the neighborhood of a given node. This does not further improve performance compared to Random Walk, as it exhibits the same disadvantages (Section 7.2.1). The DWUG Sampling approach performs well, due to its heuristical approach of connecting small clusters and selecting edges which might bear the most information, but due to its intrinsic stopping criterion might stop in a local best state. This is mitigated by random sampling new edges between nodes, thus restarting the sampling (Section 7.2.1). In general it seems there a two factors in performance degradation:

- Unconnected Components
- Sparsity

Modified Random Walk and DWUG try to bypass both these problems by sampling nodes multiple times (Section 5.3.4 and Section 5.3.6) creating a denser graph structure. For later stages of the annotation process ($\geq$ 500 annotations and one annotator per sampled edge) most sampling strategies exhibit the same performance, which can be contributed to that most nodes have been added to the *Annotated WUG* (around 90%) and most sampling strategies from this point onwards behave similar.

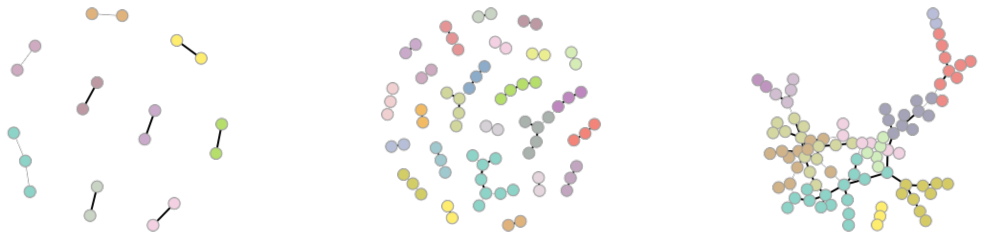

Figure 30: Random Sampled WUG with 10/50/100 annotations, illustrating the creation of unconnected components during early annotation stages.

Random Sampling Random Sampling, compared to the other sampling strategies, is the simplest strategy to execute, but this also shows in the performance, especially for small annotation sizes. The underlying problem of Random Sampling is that for small annotation sizes a lot of different unconnected components are created (Figure 30), as edges are sampled randomly and there is no intrinsic property that connects those found nodes. Hence, there is no information sampled between those nodes. The lack of information between nodes reduces the performance, as it is not possible to predict the correct clustering between those unconnected node pairs. This can be seen in both the *ARI* and *iJSD* scores. It does not matter what underlying *True WUG* is being sampled, as this behavior emerges from the sampling strategy itself. As we can observe from the Figure 29, this performance damp reduces as the annotation size grows. This is mostly due to that the number of unconnected components and the sparseness of the *Annotated WUG* reduces, as more annotations are being sampled.

Random Walk Random Walk compared to Random Sampling provides a performance enhancement for small annotation sizes, as it does not form unconnected componnets. This performance enhancement can be seen in Figure 29.

Random Walk does introduce a new problem for sparse WUGs. While traversing the WUG, it is possible that a low annotation score edge is sampled. In the case that there is no annotation error, this signals that we cross from one cluster to another. This can be a problem for sparse annotated graphs, as Figure 31 shows, because

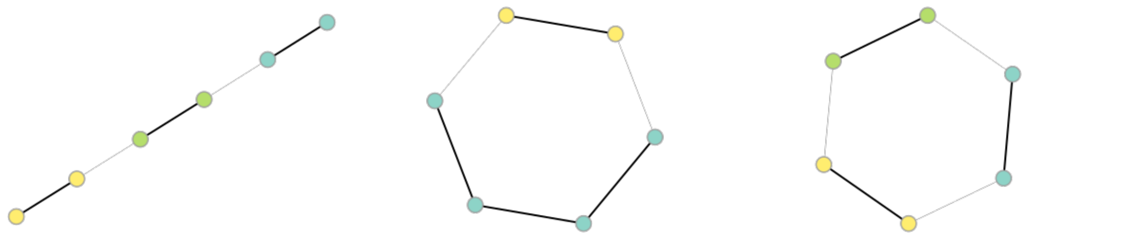

Figure 31: **Left:** Example of Random Walk at some annotation step. **Middle and Right:** Example of merging or splitting due to annotation error introduced for the newly sampled edge.

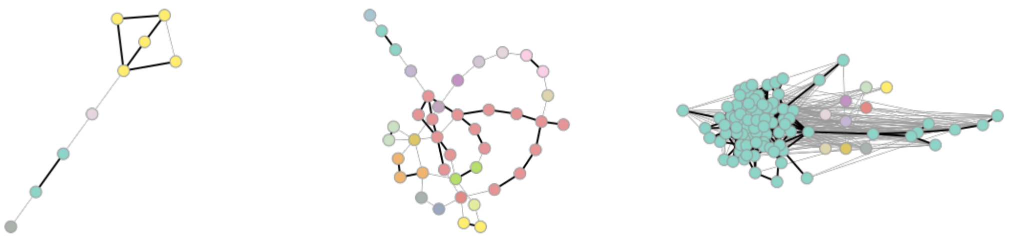

Figure 32: Random Walk with 10/50/500 annotations exemplifying the traversing of low annotation score edges and resulting multiple clustering.

if we traverse multiple low annotated edges there is no direct information if the new cluster is part of an already found cluster or not. This behaviour is amplified by the possibility of annotation error, as not only it is possible to split a cluster by annotating an edge with a low score, it is also possible that clusters may be merged (Figure 31). This effect is reduced, as more annotation are introduced into the *Annotated WUG* (Figure 32).

PageRank PageRank is an extension to the the Random Walk sampling strategy, incorporating a teleportation probability, which allows to continue the Random Walk from any possible node. Figure 33 shows that PageRank combines both undesired effects of Random Sampling and Random Walk. The effect of creating a new unconnected components is not as severe as for Random Sampling, as the frequency of restarting the Random Walk from a new node is lower, but still this behaviour is not desirable, as it reduces the performance significantly compared to Random

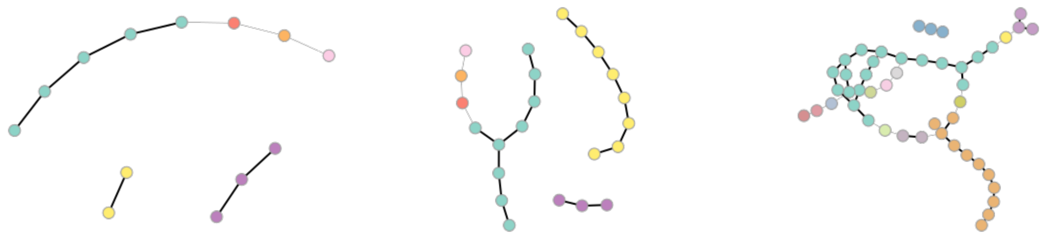

Figure 33: PageRank with 10/20/50 annotation, displaying both effects of Random Sampling (unconnected components) and Random Walk (splitting clusters) .

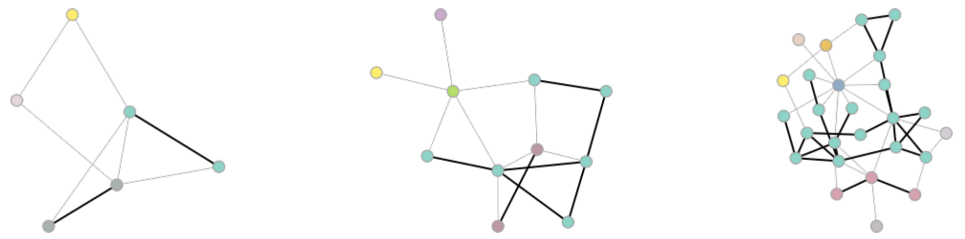

Figure 34: Modified Random Walk with 10/20/50 annotation, illustrating the dense annotation between nodes.

Walk.

Modified Random Walk This approach extends the Random Walk, by modifying how the WUG is traversed, as described in Section 5.3.4. As newly added nodes are connected to the *Annotated WUG* by two edges, each node is less susceptible to an annotation error and the resulting WUG structure is initially more densely connected when compared to other sampling strategies (Figure 34). This improves the performance drastically in the early annotation stages (Figure 29) as it reduces the probability of splitting clusters mentioned in Section 7.2.1 and does not introduces unconnected components.

Modified Search Space Random Walk MERW exhibits mostly the same performance as Random Walk, which is expected. Random Walk is mostly a special case of MERW, where only one edge of its neighborhood is sampled. Increasing the sampled neighbourhood does seem to increase the performance slightly compared to Random Walk (Figure 29), but has to be considered carefully, especially in size. Figure 14 demonstrates how sampling the neighborhood introduces stars structures in WUGs, meaning that these nodes are dependent on their parent node. Considering the effects of Random Walk observed in Section 7.2.1, we can see the susceptibility of the parent node impacts its neighbourhood significantly and thus may lead to bad performance in early annotation stages.

DWUG(+RS) Sampling The DWUG Sampling approach provides good early performance, while also exhibiting stable performance across any number of annotations. This is due to using the current clustering state of the WUG as a heuristic for edges, sampling these which might introduce the most information. For example, in the combination step of DWUG, edges are sampled for these nodes, which are not yet part of a multi-cluster and are also not connected to each cluster present in the WUG. Hence these edges are sampled as a mean of finding a possible cluster assignment for the node. During the exploration phase only edges of nodes are considered, which are yet not part of a multi-cluster but are already connected to each multi-cluster. Thus sampling these edges would introduce information about the relationship of those nodes and if they could be merged into a new cluster.

In Section 5.3.7 we explained that DWUG has an intrinsic stopping criterion, which may lead the sampling strategy in a pseudo best state. This is visible in Figure 29, as the performance stagnates for a not inconsiderable amount of annotations. By extending the DWUG Sampling strategy with Random Sampling, re-seed the sampling strategy by introducing new edges. Figure 29 shows that this approach improves the performance, as we avoid remaining in this possible pseudo state.

7.2.2 Clustering

From Figure 35 we can see that the chosen clustering strategy is detrimental for the performance of a model. We can observe that Chinese Whispers performance is highly dependent on how densely annotated a WUG is, while being robust against

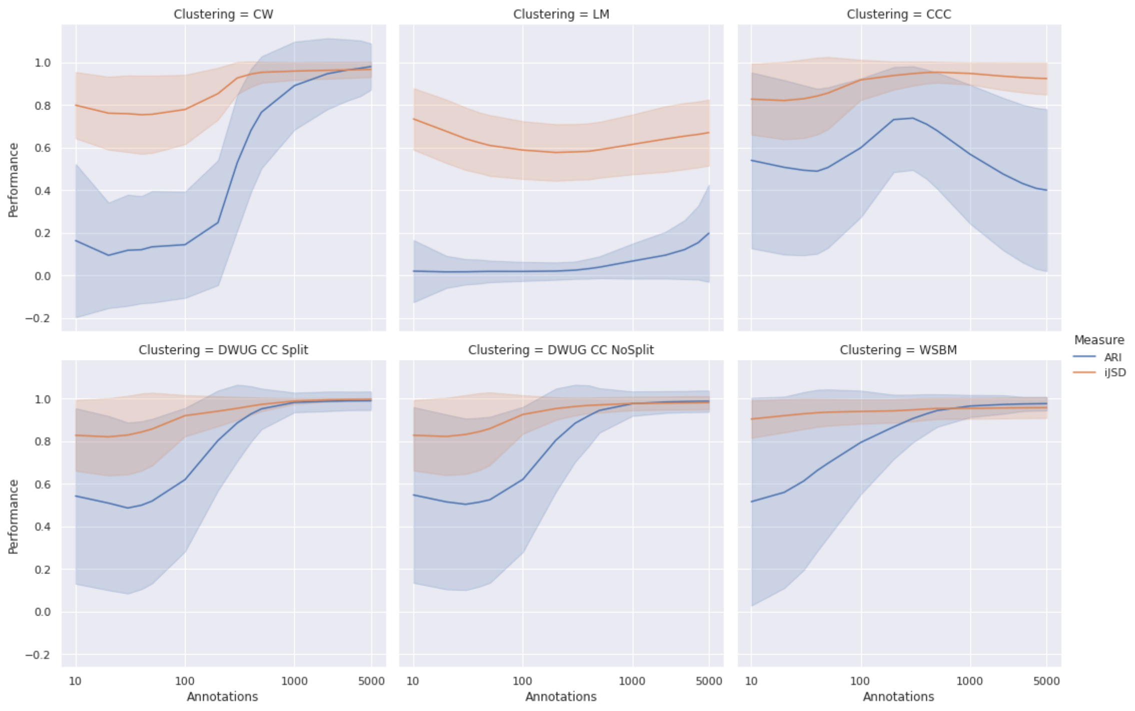

Figure 35: Overview of clustering strategies with one annotation per sampled edge. The data was generated by aggregating over all models.

annotation error (Section 7.2.2). Louvain Method's performance is unaffected by the annotation amount of a WUG, which is due to the optimization of modularity being a flawed criterion for WUGs, as it favours similar sized clusters (Section 7.2.2). For Connected Components Clustering we see that it performs well for sparsely annotated WUGs, but drops in performance for dense WUGs, which is due to annotation error (Section 7.2.2). DWUG Correlation Clustering performs similar to CCC, but due to the function being optimized exceeds the performance of CCC in densely annotated WUGs, as it does not suffer from annotation error (Section 7.2.2). The Weighted Stochastic Block Model performance can be mainly attributed to the optimization against a binomial distribution of edge-weights inside and between clusters, thus restoring the distribution used to generate the underlying *True WUG* (Section 7.2.2).

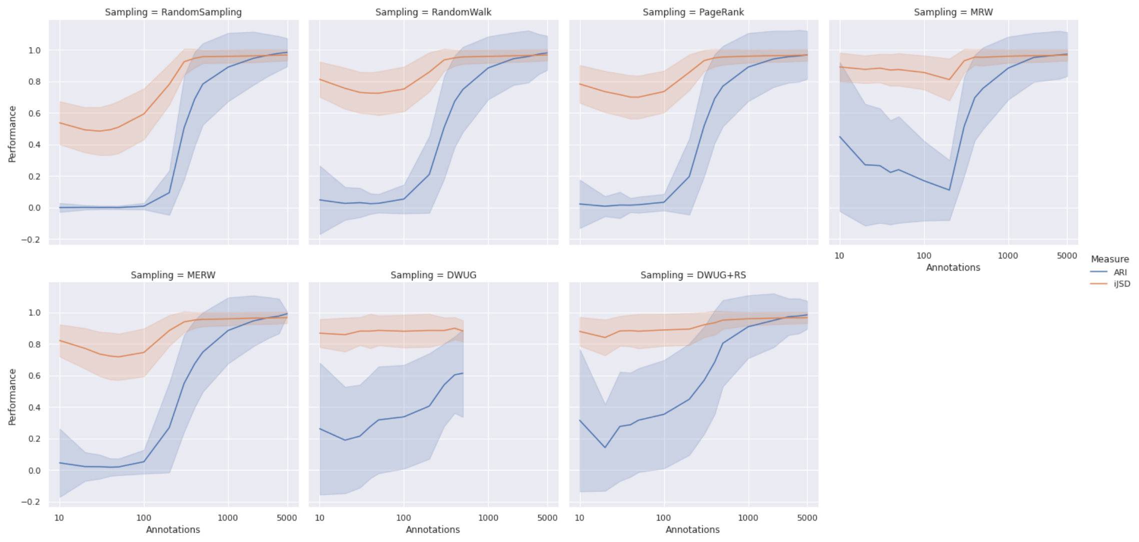

Figure 36: Overview of sampling strategies performance with one annotation per sampled edge using CW as the clustering strategy

Chinese Whispers CW exhibits poor performance for WUGs with low number of annotations (Figure 35), while performing well on highly annotated WUGs. This behaviour is due to CW only considering a nodes neighborhood during the assignment to a cluster. Hence, for sparse annotated WUGs, which are created by Random Sampling or Random Walk, the clustering can only consider one or two nodes and their cluster assignment, resulting in a locally bound optimization of the cluster assignment for a given node. As WUGs are annotated more densely the performance increases drastically, as a nodes neighborhood includes almost all nodes, thus this local optimization results in a global optimization.

CW is also robust against annotation error for densely annotated WUGs, which is due to how the cluster assignment is determined, as it considers the sum of all outgoing edges towards each cluster (Section 5.4.1). Therefore, if most edges are annotated correctly, edges with high annotation error do not contribute to the decision significantly, resulting in the correct assignment.

Louvain Method Louvain Methods poor performance is due to the modularity criterion being optimized. It has been found that modularity is insufficient for finding

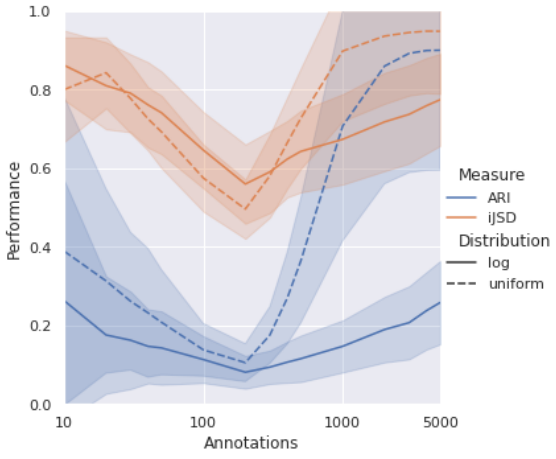

Figure 37: Comparison of Louvain Methods clustering based on uniformly and Log-distributed cluster sizes.

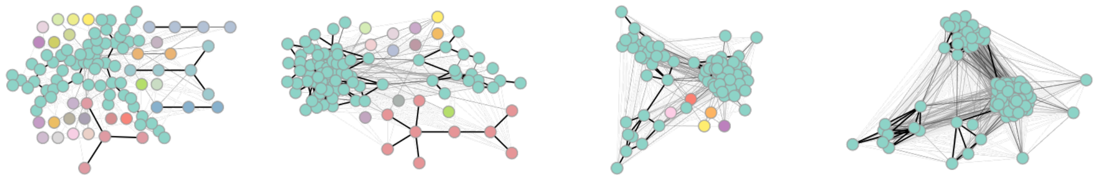

Figure 38: CCC with 200/500/1000/5000 annotation, illustrating the merging of clusters due to annotation error.

small clusters in graphs and favors a more homogeneous distribution of clusters (Fortunato and Barthelemy, 2007). This is unfavorable for the case of WUGs, as their cluster distribution tends to be skewed towards the first cluster (Figure 22 and 23 and Table 6). Figure 37 present a comparison of LM performance against a small set of WUGs, where the nodes are in one case distributed uniformly and in the other follow a logarithmic distribution. We observe that for the uniformly distributed, the performance is significantly better, while for the Log-distribution we observe the same performance as for the *Coarse WUGs*.

Connected Component Clustering Connected Component Clustering shows good performance for small number of annotations. This is due to clustering nodes

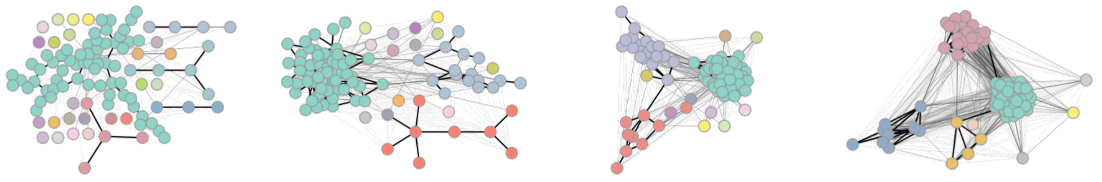

Figure 39: DWUG CC with 200/500/1000/5000 annotation, illustrating the robustness against introduced annotation error.

together that are connected by high annotation scores, which is desirable. We do see a big drop in performance as more annotations are made, which is a sign of the biggest drawback for this clustering approach, its weakness against annotation error. As more annotations are made, the probability for high annotation error introduced grows. Thus previously unconnected components might be connected by a high annotation score edge, hence merging both clusters. This effect is visible in Figure 38.

DWUG Correlation Clustering DWUG CC's performance results from the function being optimized, as it maximizes high edge-weights inside clusters while also maximizing low edge-weights between clusters (Section 5.4.4). This is a desirable optimization approach, as it captures the semantic relatedness of word usage pairs.

We observe that DWUG CC performance is similar to that of CCC (Figure 35) for less than 1000 annotations, after which they diverge. As discussed in Section 7.2.2, CCC is not robust against annotation error, leading to merges of unrelated clusters. DWUG CC's function captures the global annotation state of the WUG (Schlechtweg et al., 2021b), resulting in a robust clustering. This difference in robustness can be observed when comparing between Figure 38 and Figure 39. Comparing DWUG CC performance against that of CW, as it only uses local information (Section 7.2.2), for low number of annotations, shows that using this global information is beneficial for the overall performance.

Weighed Stochastic Block Model WSBM clustering uses a sophisticated approach, finding the most optimal clustering by minimizing its description length

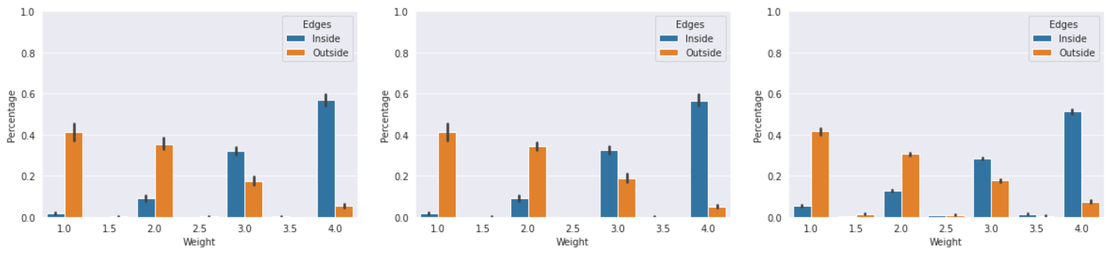

Figure 40: Weight distribution between and inside found clusters of *Fitted WUGs* with 500 annotations for DWUG CC (**Left**), DWUG CC without splitting uncorrelated clusters (**Middle**) and WSBM (**Right**)

(Peixoto, 2014a). Most notably this approach also takes into account the distribution of edge-weights and optimizes towards a clustering, such that the weights between and inside the clusters follow a given distribution. As we used this approach optimizing towards a binomial distribution, due to the observations made in Section 4.3, Section 7.1 and Figure 26, it is not surprising that this approach performs well.

Figure 40 shows the distribution found by the DWUG CC and WSBM approach on *Fitted WUGs*. It is interesting to observe, that both approaches fit the binomial distribution rather well, especially in the case of DWUG CC as it does not optimize for this behaviour intrinsically. As we will discuss in Section 7.3, the DWUG CC approach does not perform as well as WSBM for a low number of annotations. Therefore this shows that optimizing towards this behaviour is not the only characteristic that should be considered for clustering approaches.

It is important to note, that we used WSBM to generate the coarse and *Fitted WUGs*, thus the results are biased.

7.2.3 Stopping Criterion

Stopping criterions are important deciding factors when to stop the annotation process and thus are crucial for an efficient and effective model. Figure 41 shows that the Bootstrapped JSD (here inverse) does not capture any important information about

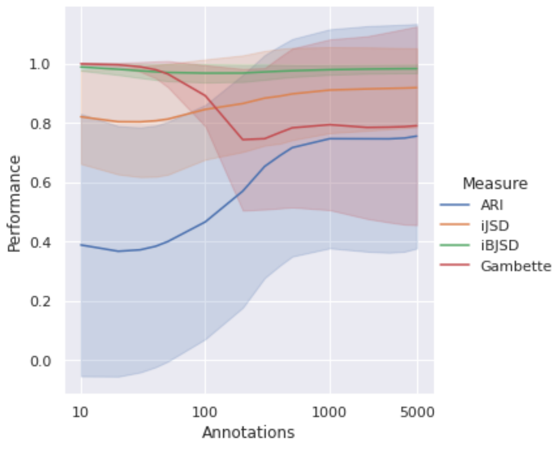

Figure 41: Overview comparison of stopping criterions against measured performance.

the *Annotated WUG* and thus is not a usable indicator. Gambettes method does capture information about the annotation process, but only produces reasonable statements after 200 annotations, thus being only partially usable.

Bootstrapped JSD From Figure 41, we can see that the Bootstraped iJSD metric does not correlate to the observed iJSD, nor captures any information about the *Annotated WUG* and stays close to 1. This is due to how the metric is calculated. By random sampling with replacement from a distribution, we expect the sampled distribution to be closely related to the actual underlying distribution. As we bootstrap the random samples from the cluster distribution of the *Annotated WUG*, we expect a similar cluster distribution, resulting in a JSD ≈ 0 (iJSD ≈ 1) and thus the resulting percentile value to be close to 0 (iJSD ≈ 1) as well.

Gambette Gambettes measured performance follows the actual ARI very closely from around 200 annotations (Figure 41), indicating that this stopping criterion can be used very well for approximating the actual performance of the *Annotated WUG*. The problem for sparse annotated WUGs arises from the number of randomly annotated edges. Gambettes random annotator only annotates already existing edges

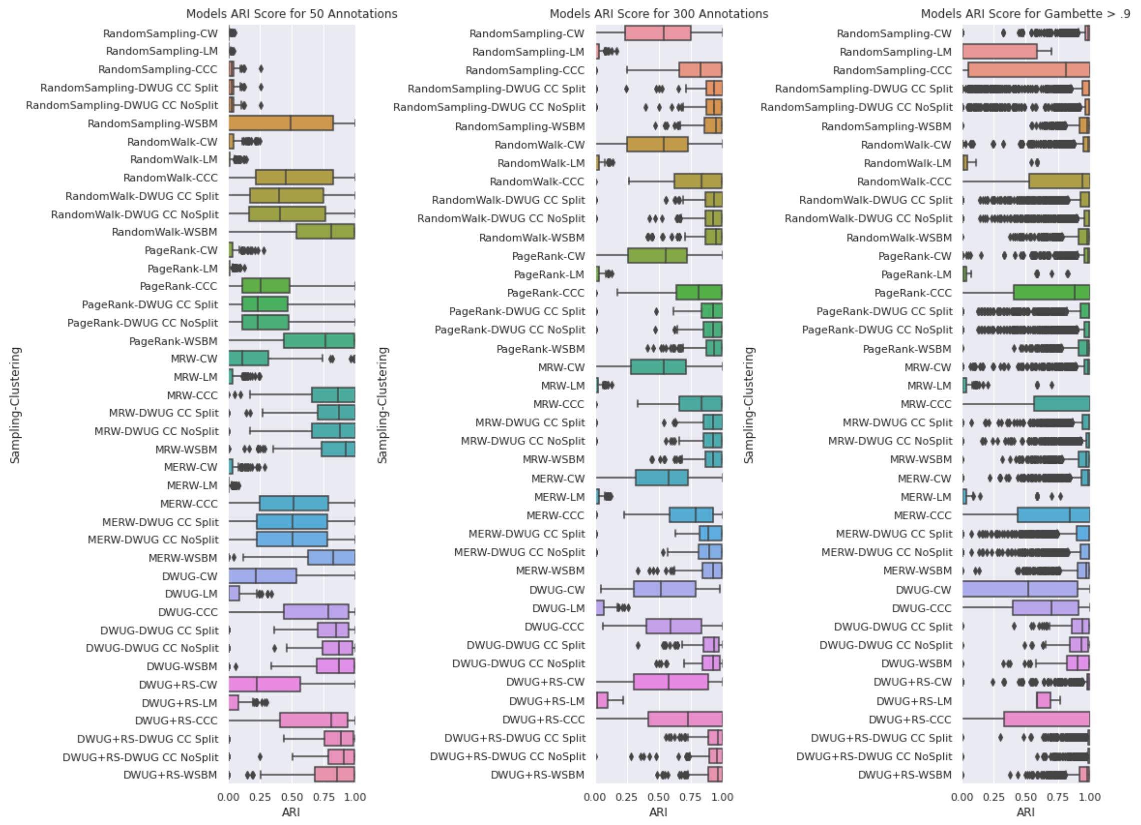

Figure 42: Models on *Coarse WUGs* in comparison for ARI Score based on the number of annotations with 50 annotations (**left**), with 100 annotations (**Middle**) and with Gambette > .9 and at least 100 annotations.

in the *Annotated WUG*, resulting in an additional annotation of this edge, hence the probability of heavily influencing the weight of this edge is low. Additionally, only 10% of the existing edges are annotated, consequently only a few edges are annotated by the random annotator. Thus the newly clustered permuted WUG is very similar to the initial *Annotated WUG*.

7.2.4 Model Overview

Figure 42 shows the performance of different models at 50 and 300 annotations. We only include the scores for ARI, as iJSD behaves relatively similar. We choose to compare the models at 50 annotations to understand how well the models perform under low number of annotations and at 300 annotations as this captures the most common number of annotations found in the observed data (Section 4.3). For higher number of annotations, sampling strategies perform similar (Figure 29) and a models performance only depends on the clustering strategy employed. Hence for these cases we would recommend DWUG CC or WSBM as they identify the correct clustering very well and are robust against annotation noise (Section 7.2.2 and Section 7.2.2). CW should also be considered for later annotation stages, as its robust against annotation error, providing good performance and lower time complexity.

Figure 42 for models with 50 annotations illustrates the favourable trait of building dense structures and sampling edges based on some heuristics, as MRW and DWUG (+RS) sampling exhibit good performance. It is important to note, that MRW in contrast to DWUG (+RS) sampling does not depend on a clustering strategy during the annotation phase, thus being more fitted for environments where clustering is not possible during this phase.

Figure 42 for models with 300 annotations demonstrate the importance of clustering strategies incorporating and optimizing towards characteristics found in WUGs, as these clustering strategies, namely DWUG CC and WSBM, exhibit higher and stable performance compared to other strategies.

We can see that models using MRW or DWUG (+RS) in combination with DWUG CC or WSBM achieve higher performance with less annotations than other models, which is in line with the observations made in the previous sections.

In Figure 42 (right) we include a possible use case of Gambette, where the annotation process is stopped after the calculated mean ARI of Gambette is above the threshold of 0.9. For most models this stopping criterion is triggered between 500 and 1000 annotations, which is in line with the performance observations made in previous sections, hence pointing towards the usefulness of this criterion.

7.2.5 Increasing Annotators per Edge

Annotation noise can introduce undesired effects, like splitting clusters (Section 7.2.1, 7.2.1 and 7.2.1) or merging unrelated clusters in the case of CCC (Section

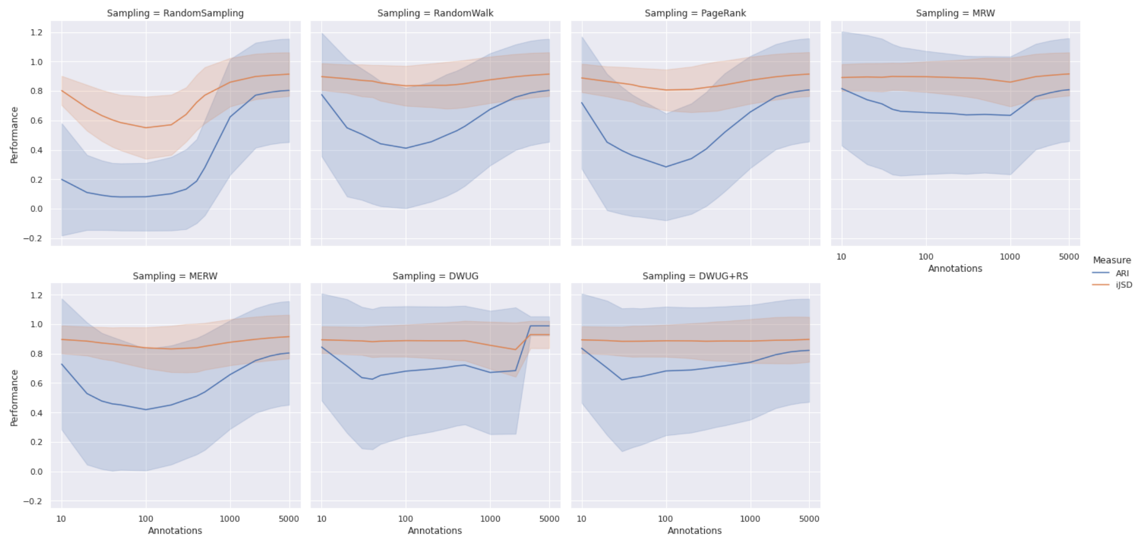

Figure 43: Overview of sampling strategies with five annotation per sampled edge. The data was generated by aggregating over all models.

7.2.2). Table 4 and Figure 7 show that most annotation error lies between 0 and .5, hence annotations mostly deviate by one on the DURel relatedness scale (Table 1). Therefore, introducing more annotators per edge, we would expect to dampen the effect high of annotation error, particularly for one annotator. As clustering strategies are especially susceptible to noise, we compare the performance of clustering strategies with 5 annotators per edge (Figure 44) against the performance with 1 annotator per edge (Figure 35), and observe that the performance for DWUG CC and CCC rises significantly. Increasing the annotation load per edge also increases the annotations needed to annotate the same amount of edges. This is visible in the shift of performance for sampling strategies (Figure 43). The high performance for low number of annotations ($\approx$ 10) is a spurious effect, due to the low number of edges sampled. The trade off between performance and annotation load has to be considered carefully.

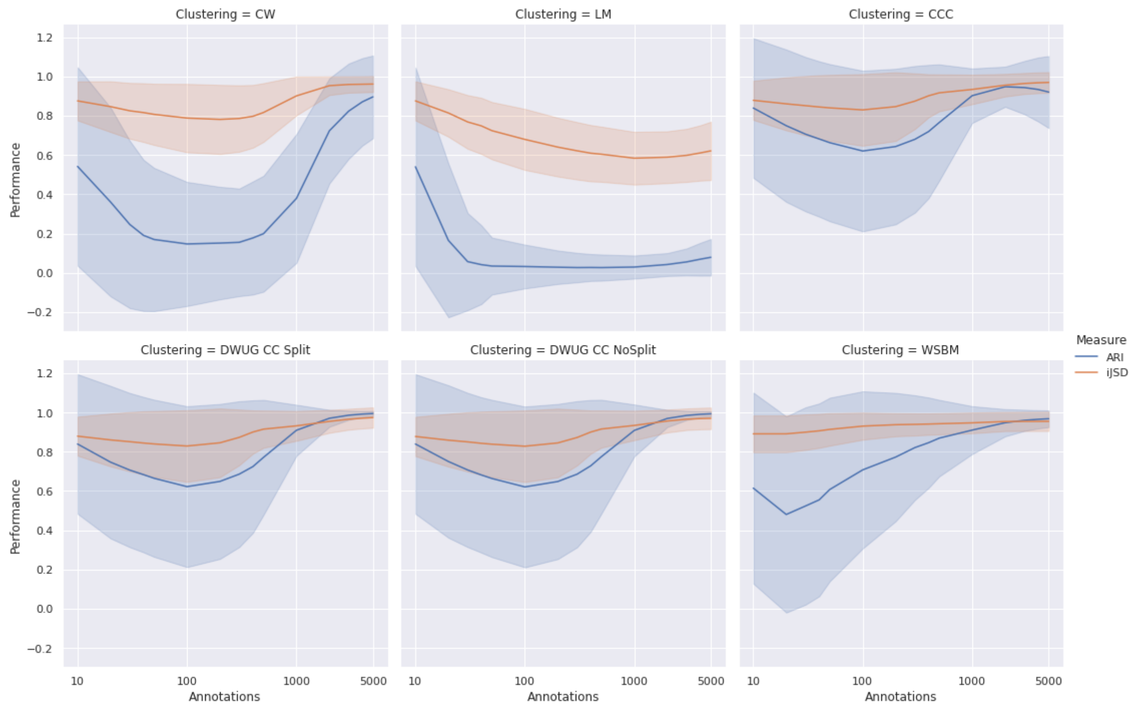

Figure 44: Overview of clustering strategies with five annotation per sampled edge. The data was generated by aggregating over all models.

7.3 Models on Fitted WUGs

In Section 7.1, we discussed the differences between observed, *Coarse* and *Fitted WUGs*. The main difference between *Fitted WUGs* and *Coarse WUGs* are the weight and annotation distributions (Figure 26 and Figure 27), where the *Fitted WUGs* follow the observed distribution more closely, hence the distinction between unrelated and related pairs becomes more blurred. Considering annotation error, the probability for annotating an unrelated pair as being related rises. This reflects heavily in the performance of each model (Figure 45), as clustering strategies are not able to distinguish the cluster assignment of a node as clearly as in the case for *Coarse WUGs*, particularly for small number of annotations. We still observe that the sampling strategies MRW and DWUG paired with DWUG CC and WSBM perform better than other models.

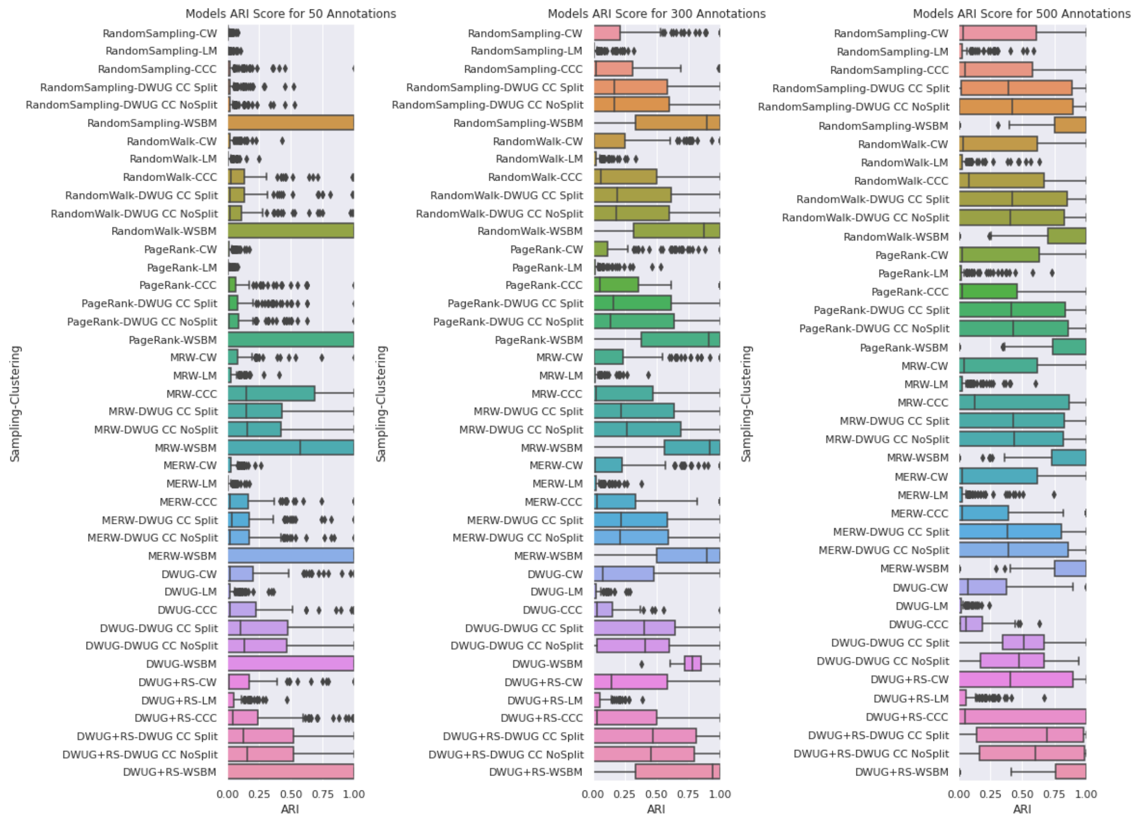

Figure 45: Models on *Fitted WUGs* in comparison for ARI score based on the number of annotations. **Left** 50, **middle** 300 and **right** 500 annotations.

8 Conclusion

The goal of this thesis was to test different models, consisting of a sampling, clustering and stopping strategy, exhaustively on their ability to efficiently and effectively find the correct sense assignment of word usages in WUGs. Hence we build a framework, which is able to simulate the real world annotation process of WUGs, allowing the models to be evaluated and compared extensively and automatically without the need for human annotators.

In Section 4 we analyzed different characteristics exhibited by the WUGs in DWUG DE/EN and DiscoWUG data set, including observations made about their structure as well as their annotations. In Section 7.1 we showed that it is possible to generate WUGs with characteristics resembling that of observed WUGs, based on the observations made. This allowed us to use these as underlying *True WUGs* in the simulation and evaluation of performance for different models, giving credibility to their results.

Section 7.2 and Section 7.3 discussed the various advantages and disadvantages of different sampling, clustering and stopping strategies. We found that models using Modified Random Walk (Section 5.3.4) or DWUG Sampling (Section 5.3.6) in combination with DWUG Correlation Clustering (Section 5.4.4) or Weighted Stochastic Block Model (Section 5.4.5) clustering strategy achieved good results, even for very low number of annotations. This is due to these models generating dense structures as well as optimizing towards characteristics observed in the data set. Furthermore, we found that Gambettes Method (Section 5.5.2) provides a good approximation of how well the *Annotated WUG*, generated by a model, captures the correct cluster assignment of nodes.

It is interesting to note that these models can be seen as solutions to an optimization problem. For example a sampling strategy has to find an optimal ordering of edges, such that the edges which provide the most information are added earlier in the annotation process. Thus, strategies should be considered which are also effective for other optimization problems.

There are two drawbacks in our current approach. As we simulated on a diachronic data set, meaning that each WUG contained information from two time periods, without differentiating between those, our results might be biased towards such data sets. However, we analyzed the time periods separately (Section 4) and found that they exhibit similar characteristics, suggesting that these results might be generalizable. It is also important to note, that we did not further improve the parameters of a model. This may lead to better results, but may also be highly dependent on the underlying *True Graphs.*

We did not further research the possibility of using the triangular property (Section 4.3) to enhance the models, but think that this is an important topic to further delve into. As we have shown that it is possible to generate WUGs and simulate the annotation process, this may serve as a testing ground for defining and refining models before they are used in a real world application.

References

Christopher Aicher, Abigail Z. Jacobs, and Aaron Clauset. Learning latent block structure in weighted networks. *Journal of Complex Networks*, 3(2):221—-248, Jun 2014. ISSN 2051-1329. doi: 10.1093/comnet/cnu026. URL `http://dx.doi.org/10.1093/comnet/cnu026`.

Nikhil Bansal, Avrim Blum, and Shuchi Chawla. Correlation clustering. *Machine Learning*, 56(1-3):89–113, 2004. doi: 10.1023/B:MACH.0000033116.57574.95.

Chris Biemann. Chinese whispers - an efficient graph clustering algorithm and its application to natural language processing problems. In *Proceedings of TextGraphs: the First Workshop on Graph Based Methods for Natural Language Processing*, pages 73–80, New York City, June 2006. Association for Computational Linguistics. URL `https://aclanthology.org/W06-3812`.

Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, oct 2008. doi: 10.1088/1742-5468/2008/10/p10008. URL `https://doi.org/10.1088/1742-5468/2008/10/p10008`.

Jaewon Chung, Benjamin D Pedigo, Eric W Bridgeford, Bijan K Varjavand, Hayden S Helm, and Joshua T Vogelstein. Graspy: Graph statistics in python. *J. Mach. Learn. Res.*, 20:158–1, 2019.

B. Efron. Bootstrap Methods: Another Look at the Jackknife. *The Annals of Statistics*, 7(1):1 – 26, 1979. doi: 10.1214/aos/1176344552. URL `https://doi.org/10.1214/aos/1176344552`.

D.M. Endres and J.E. Schindelin. A new metric for probability distributions. *IEEE Transactions on Information Theory*, 49(7):1858–1860, 2003. doi: 10.1109/TIT.2003.813506.

Katrin Erk, Diana McCarthy, and Nicholas Gaylord. Investigations on word senses and word usages. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1*, pages 10–18, Stroudsburg, PA, USA, 2009.

Katrin Erk, Diana McCarthy, and Nicholas Gaylord. Measuring word meaning in context. *Computational Linguistics*, 39(3):511–554, 2013.

Santo Fortunato and Marc Barthelemy. Resolution limit in community detection. *Proceedings of the national academy of sciences*, 104(1):36–41, 2007.

Philippe Gambette and Alain Guénoche. Bootstrap clustering for graph partitioning. *RAIRO - Operations Research - Recherche Opérationnelle*, 45(4):339–352, 2011. doi: 10.1051/ro/2012001.

Paul W. Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social Networks*, 5(2):109 – 137, 1983.

John Hopcroft and Robert Tarjan. Algorithm 447: Efficient algorithms for graph manipulation. *Commun. ACM*, 16(6):372–378, June 1973. ISSN 0001-0782. doi: 10.1145/362248.362272. URL `https://doi.org/10.1145/362248.362272`.

Adam Kilgarriff. "I don't believe in word senses". *Computers and the Humanities*, 31(2), 1997.

Sinan Kurtyigit, Maike Park, Dominik Schlechtweg, Jonas Kuhn, and Sabine Schulte im Walde. Lexical semantic change discovery, 2021a.

Sinan Kurtyigit, Maike Park, Dominik Schlechtweg, Jonas Kuhn, and Sabine Schulte im Walde. Lexical Semantic Change Discovery. In *Proceedings of the Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language*

Processing, Online, 2021b. Association for Computational Linguistics. URL `https://arxiv.org/abs/2106.03111`.

Diana McCarthy, Marianna Apidianaki, and Katrin Erk. Word sense clustering and clusterability. *Computational Linguistics*, 42(2):245–275, 2016.

Tiago P. Peixoto. Efficient monte carlo and greedy heuristic for the inference of stochastic block models. *Physical Review E*, 89(1), Jan 2014a. ISSN 1550-2376. doi: 10.1103/physreve.89.012804. URL `http://dx.doi.org/10.1103/PhysRevE.89.012804`.

Tiago P. Peixoto. The graph-tool python library. *figshare*, 2014b. doi: 10.6084/m9.figshare.1164194. URL `http://figshare.com/articles/graph_tool/1164194`.

Tiago P. Peixoto. Nonparametric weighted stochastic block models. *Physical Review E*, 97, 08 2017. doi: 10.1103/PhysRevE.97.012306.

William M Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

Dominik Schlechtweg, Sabine Schulte im Walde, and Stefanie Eckmann. Diachronic Usage Relatedness (DURel): A framework for the annotation of lexical semantic change. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 169–174, New Orleans, Louisiana, 2018. URL `https://www.aclweb.org/anthology/N18-2027/`.

Dominik Schlechtweg, Barbara McGillivray, Simon Hengchen, Haim Dubossarsky, and Nina Tahmasebi. SemEval-2020 Task 1: Unsupervised Lexical Semantic Change Detection. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain, 2020. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/2020.semeval-1.1/`.

Dominik Schlechtweg, Enrique Castaneda, Jonas Kuhn, and Sabine Schulte im Walde. Modeling sense structure in word usage graphs with the weighted stochastic block model. In *Proceedings of *SEM 2021: The Tenth Joint Conference* *on Lexical and Computational Semantics*, pages 241–251, Online, August 2021a. Association for Computational Linguistics. doi: 10.18653/v1/2021.starsem-1.23. URL `https://aclanthology.org/2021.starsem-1.23`.

Dominik Schlechtweg, Nina Tahmasebi, Simon Hengchen, Haim Dubossarsky, and Barbara McGillivray. DWUG: A large Resource of Diachronic Word Usage Graphs in Four Languages. *CoRR*, abs/2104.08540, 2021b. URL `https://arxiv.org/abs/2104.08540`.

Dmitry Ustalov, Alexander Panchenko, Chris Biemann, and Simone Paolo Ponzetto. Watset: Local-Global Graph Clustering with Applications in Sense and Frame Induction. *Computational Linguistics*, 45(3):423–479, 2019. ISSN 0891-2017. doi: 10.1162/COLI_a_00354.

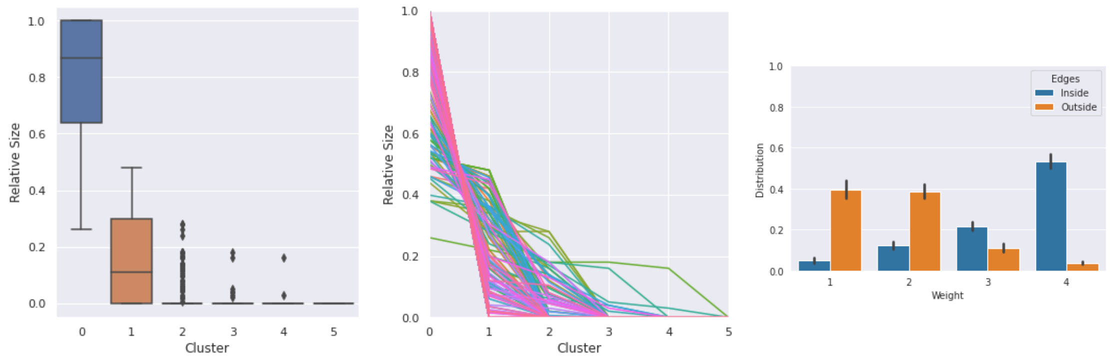

Figure 46: **Left:** Relative cluster size for inferred graphs. **Middle:** Relative cluster size per inferred graph. **Right:** Weight distribution for inferred graphs.

Appendices

A Inferred Graphs Characteristics

As the WSBM is used to infer the parameters for *Fitted WUGs* from the data set DWUG DE/EN and DiscoWUG (Section 6), we also include Figure 46, showing that these inferred graphs also capture the characteristics observed in Section 4.

B German Summary

B.1 Einleitung

Wortverwendungsgraphen (WUGs, McCarthy et al., 2016; Schlechtweg et al., 2021b) sind ein Ansatz zur Darstellung von Beziehungen zwischen Wortverwendungspaaren, wobei jede Wortverwendung als Knoten dargestellt wird und die gewichtete ungerichtete Kante zwischen einem solchen Paar die semantische Nähe darstellt. Schlechtweg et al. (2020) benutzt WUGs um semantische Änderung zu erkennen, während McCarthy et al. (2016) diese Darstellung ausnutze, um durch Clustering-Prozeduren Sinne von Wörtern zu erkennen. Diese Darstellung hat ihren Nachteil, da für eine gute Sinneserkennung im Wortverwendungsgraphen ein vollständiger Graph von Nöten wäre. Da aber die Anzahl der zu annotierenden Kanten quadratisch mit der Anzahl der Knoten in einem WUG wächst, würde sich dies auf den Annotationsprozess Negativ auswirken.

Unser Ziel ist es, effizienten und effektiven Modelle zu finden, die aus einer Sampling und Cluster-Strategie sowie einem Stoppkriterium bestehen. Dies erreichen wir durch die Evaluierung von Modellen in einer Simulationsumgebung, die ein generatives Verfahren verwendet, um automatisiert vollständige WUGs zu generieren und auf Basis dieser die Modelle zu simulieren und evaluieren. Die Basis dafür bietet der Deutsche und Englische DWUG Datensatz (Version 1.0.0) (Schlechtweg et al., 2021b) und der DiscoWUG Datensatz (Version 1.0.0) (Kurtyigit et al., 2021a).

B.2 Wortverwendungsgraph Analyse

Bevor wir uns der Simulation widmen, analysieren wir die derzeitigen existierenden, von Menschen annotierten Wortverwendungsgraphen. Hierfür beleuchten wir verschiedene Aspekte dieser WUGs, wie die Anzahl der Wortverwendungen, Sinnesstrukturen und derren Annotationen. Die Anzahl der Wortverwendungen ist wichtig zu behandeln, da wenn es möglich ist, ein WUG vollständig zu annotieren, der Einsatz von Sampling-Strategien nicht begründbar ist. Die Sinnesstrukturen zu verstehen ist deshalb wichtig, um geeignete simulierte WUGs zu generieren und sowie gute Ansätze zu finden, die die zugrunde liegenden Merkmale eines Wortsinnes erfassen. Um einen realistischen Annotationsprozess zu simulieren, ist es wichtig zu analysieren, wie Annotationen und Annotatoren sich untereinander unterscheiden.

Wortverwendung Da die Anzahl der Annotationen quadratisch mit den Wortverwendungen in einem WUG anwächst, beobachten wir, dass sogar für die kleinsten WUGs im Datensatz der Annotationsprozess außerhalb des möglichen Rahmens liegt.

Sinne Für die Sinne in den WUGs erkennen wir, dass diese grob einer logarithmischen Normalverteilung ähneln. Dies stimmt mit der Beobachtung von Kilgarriff (1997) überein. Weiterhin beobachten wir, dass die Sinnesstrukturen für einzelne WUGs eher einer Poisson-Verteilung ähneln.

Annotation Die annotierten WUGs besitzen meist um die 500 Annotationen. Dies ist interessant, da somit die meisten Graphen nur eine sehr geringe Annotationsdichte besitzen. Auch interessant zu sehen ist, dass der Annotationsfehler eine nicht unerhebliche Konsequenz des Annotationsprozesses ist. Dies spiegelt sich besonders in der Varianz der Annotation (um die .22), sowie in der Anzahl der Null-Annotationen auf der DURel-Skala (Schlechtweg et al., 2018) (4.4%) wieder.

B.3 Simulation

Da das Ziel der Simulation ist, das Annotationsverfahren von WUGs realisitsch zu reproduzieren, nutzen wir die folgenden zwei Graphen. Den **Wahren WUG** und den **Annotierten WUG**. Der *Wahre WUG* ist dabei ein vollständiger Graph, bei der die Kantengewichte und die entsprechende Cluster-Zuweisung die korrekte Darstellung eines WUGs für ein nicht spezifiziertes Wort darstellen. Basierend auf diesem *Wahren WUG* führen die verschiedenen Modelle ihre Sampling, Cluster und Stop-Prozeduren aus und generieren damit den *Annotierten WUG*. Für die *Wahren WUG*s generieren wir zwei Datensätze. **Groben WUG**s und **Angepassten WUG**s. Diese unterscheiden sich insofern, dass für die *Groben WUG*s nur observierte Charakteristiken für den Generierungsprozess verwendet wurden, während für die *Angepassten WUG*s der Datensatz selber verwendet wurde.

Graph Generierung Um diese *Wahren WUG*s in der Simulation verwenden zu können, bedienen wir uns der Technik von Schlechtweg et al. (2021a). Hierbei wird ein generativer Prozess verwendet, das Weighted Stochastic Block Model (Aicher et al., 2014; Peixoto, 2017), um mögliche Graphen generieren zu können. Schlechtweg et al. (2021a) zeigt, dass es mit diesem Verfahren möglich ist, Graphen zu generieren, die den beobachteten Graphen sehr nahe kommen. Dies erlaubt uns *Wahre WUG*s für unsere Simulation zu generieren, ohne auf Annotatoren zurückgreifen zu müssen.

Annotation Da wir beobachten konnten, dass das Annotationsverfahren zu nicht unerheblichen Fehlannotation auf dem WUGs führen, simulieren wir folgende zwei Prozesse bei der Annotation einer Kante. Der Annotationsfehler modelliert die Abweichung der Annotation von der "wahren" Annotation, indem es einen Fehler mithilfe einer Verteilung berechnet und diesen mit verrechnet. Eine Null-Annotation beschreibt die Unentschlossenheit eines Annotators und wird zufällig mit einer gewissen Wahrscheinlichkeit mit der "wahren" Annotation ersetzt.

Modelle Modelle bestehen aus drei Grundkomponenten, ein Sampling, Cluster und Stop-Prozedur. Wir definieren mehrere Komponenten und bilden aus denen verschiedene Modelle, welche wir dann auf ihre Leistungsfähigkeit prüfen.

B.4 Analyse

Graph Generation Es ist wichtig, die Eigenschaften der generierten WUGs zu analysieren, um festzustellen, wie gut die *Groben WUG*s und die *Angepassten WUG*s mit den beobachteten WUGs vergleichbar sind und diese erfassen, um die Glaubwürdigkeit der Modelle und deren Einsatz in nicht simulierten Annotationsprozessen zu untermauern. Hierbei können wir feststellen, dass wir die beobachteten Eigenschaften sehr gut annähren können, sowohl für die Sinnesverteilung als auch die Annotationsfehler.

Sampling-Prozeduren Im Allgemeinen scheint es zwei Faktoren zu geben, die zu einer Leistungsminderung der Sampling-Prozedur führt. Die unverbundenheit von Komponenten sowie dünnbesetzte Graphen. Dies wird besonders deutlich, wenn wir uns die Leistungen der Prozeduren für wenige Annotationen betrachten. Prozeduren, welche Knoten-Paare generieren, die nicht zum restlichen Graph verbunden sind, oder welche, die eine geringe Dichte von Kanten bilden, schneiden im Vergleich zu anderen, welche genau dieses Verhalten umgehen, schlechter ab.

Cluster-Prozeduren Die Cluster-Prozedur ist für die Leistung eines Modells äußerst wichtig. Dabei beobachten wir, dass die Leistung abhängig von mehreren Faktoren ist. Unter anderem ist die gewählte Sampling-Prozedur wichtig, da diese sich stark auf das erkennen von Clustern auswirken kann. Dabei sehen wir, dass besonders Prozeduren, welche für wenige Annotationen unverbundene Komponenten generieren, besonders schlecht abschneiden. Weiterhin beobachten wir das die Grundstruktur des Graphens, besonders die Cluster-Größe des *Wahren WUG*s, sich stark auf die Leistung auswirkt. Wir können außerdem erkennen, dass Cluster-Prozeduren, welche nicht robust gegen Annotationsfehlern sind, ebenfalls schlechte Ergebnisse liefern.

Stop-Kriterien Abbruchkriterien sind wichtige Entscheidungsfaktoren dafür, wann der Annotationsprozess beendet werden kann, und sind daher für ein effizientes und effektives Modell von entscheidender Bedeutung. Wir haben zwei Stop-Kriterien untersucht, *Bootstrapped JSD* und *Gambette*. Dabei konnten wir erkennen, dass *Bootstrapped JSD* keine wichtigen Informationen über den *Annotierten WUG* erfasst und daher kein brauchbarer Indikator ist. *Gambette* erfasst zwar Informationen über den Annotationsprozess, aber erst nach genügend Annotationen. Was aber interessant ist, ist das *Gambette* die Leistung des *Annotierten WUG* sehr gut annähren kann und ist deshalb als Abbruchkriterien geeignet.

Leistung der Modelle Wir konnten beobachten, dass Modelle, wo die einzelnen Komponenten die oben genannten Faktoren beachten, sehr gute Leistungen auch für sehr niedrige Annotationen produzieren. Die Modelle schneiden aber bei den *Angepassten WUG*s deutlich schlechter ab, da die Gewichtsverteilung sich zwischen den *Groben WUG*s und *Angepassten WUG*s stark unterscheidet.

B.5 Fazit

Das Ziel dieser Arbeit war, verschiedene Modelle, bestehend aus einer Sampling-, Cluster- und Stop-Prozedur, umfassend auf ihre Fähigkeit zu testen. Dabei lag das

Hauptmerkmal auf der korrekten Sinnzuweisung von Wortverwendungen in WUGs effizient und effektiv zu finden. Modelle, die sowohl dichte Strukturen erzeugen als auch auf die im Datensatz beobachteten Merkmale hin optimieren, erbringen gute Leistungen. Wir haben ebenfalls gezeigt, dass es möglich ist, *Wahre WUG*s mithilfe des generativen Modelles zu erstellen, deren Eigenschaften denen der beobachteten WUGs (DWUG DE/EN und DiscoWUG) ähnelt. Dies erlaubt es uns, WUGs für die Simulation als auch für die Bewertung der Leistung der verschiedenen Modelle zu verwenden und verleiht deren Ergebnissen Glaubwürdigkeit.

Erklärung

Ich versichere, diese Arbeit selbstständig verfasst zu haben. Ich habe keine anderen als die angegebenen Quellen benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommene Aussagen als solche gekennzeichnet. Weder diese Arbeit noch wesentliche Teile daraus waren bisher Gegenstand eines anderen Prüfungsverfahrens. Ich habe diese Arbeit bisher weder teilweise noch vollständig veröffentlicht. Das elektronische Exemplar stimmt mit allen eingereichten Druck-Exemplaren überein.

Datum und Unterschrift: 29.10.21 [illegible]

Declaration

I hereby declare that the work presented in this thesis is entirely my own. I did not use any other sources and references that the listed ones. I have marked all direct or indirect statements from other sources contained therein as quotations. Neither this work nor significant parts of it were part of another examination procedure. I have not published this work in whole or in part before. The electronic copy is consistent with all submitted hard copies.

Date and Signature: 29.10.21 [illegible]