Sense Tree: Discovery of New Word Senses with Graph-based Scoring

Jan Ehmüller[1], Lasse Kohlmeyer[1], Holly McKee[1], Daniel Paeschke[1], Tim Repke[2], Ralf Krestel[2], and Felix Naumann[2]

Hasso Plattner Institute, University of Potsdam, Germany
[1]first.last@student.hpi.uni-potsdam.de,[2]first.last@hpi.uni-potsdam.de

Abstract. Language is dynamic and constantly evolving: both the usage context and the meaning of words change over time. Identifying words that acquired new meanings and the point in time at which new *word senses* emerged is elementary for *word sense disambiguation* and *entity linking* in historical texts. For example, *cloud* once stood mostly for the weather phenomenon and only recently gained the new sense of *cloud computing*. We propose a clustering-based approach that computes *sense trees*, showing how meanings of words change over time. The produced results are easy to interpret and explain using a drill-down mechanism. We evaluate our approach qualitatively on the Corpus of Historic American English (COHA), which spans two hundred years.

1 The Evolution of Language

As language evolves, words develop new senses. Detecting these new senses over time is useful for several uses, such as *word sense disambiguation* and *entity linking* in historic texts. Current machine learning models that represent and disambiguate word senses or link entities are trained on recent datasets. Therefore, these models are not aware of how language changed over the last decades or even centuries. When disambiguating words in historic texts, it is not possible for such a model to know which senses a word could have had at that time. For instance, the word *cloud* was once used mostly in the newspaper section of weather forecasts. Nowadays, it is increasingly used in the context of *cloud computing*. Hence, when disambiguating *cloud* in texts from the 19th century with a model trained on current data, the model would not be aware that the meaning of *cloud computing* did not yet exist at that point in time.

New word senses enter colloquial language from many aspects of human life, such as popular culture, new technologies, world events, and social movements. In 2019 alone, over 1,100 new words and meanings were added to the Merriam-Webster Dictionary. Because the usage of context words differs over time, we can discover word senses by taking advantage of this temporal aspect. Discovering new or different senses of a word is known as word sense induction (WSI). An example for WSI can be found in the context of *depression*. *Depression* is commonly known in the sense of a mental health disorder, but also in the sense

of economic crisis in the term *great depression*. The aim of WSI is to find out which senses the word *depression* has. Word sense disambiguation (WSD), on the other hand, is known as the automated disambiguation of a word sense within a text [15]. While WSI aims to discover different senses for one word, WSD aims to decide which sense a word has in a specific context, such as in a sentence.

We propose a WSI method that detects new senses by creating multiple co-occurrence graphs over time, and extracts word senses based on so-called ego-networks. For a given word, it uses graph clustering to extract word senses from the word's ego-network. These word senses are matched over time to create a forest of so-called "sense trees". This forest can be explored to find out if and when a word gained new senses over time. We evaluated our approach using the Corpus of Historical American English (COHA), with 400 million words the largest corpus of its type [4].

2 Related Work

Research on word sense induction and word sense disambiguation addresses how to improve information retrieval in search engines, information extraction for specific domains, entity linking over time, machine translation, and lexicography [15]. Our research focuses on WSI and aims to discover the emergence of new word meanings over time. Approaches can be divided roughly into vector space models that include word embeddings [5], graph clustering techniques [19], which include word clustering [5], and co-occurrence graphs [14].

Word embeddings are vector representations of words in a semantic space that allow for word meanings to be induced by context. Word embeddings are trained by a neural network that learns to predict a word based on its context words or vice versa. Hence, words with similar contexts have a similar vector and are thus in close proximity to each other. The initially proposed word2vec model [13] computes only one vector and thus, allows for only one meaning per word. Natural language is ambiguous and most words are polysemous – they have multiple senses. Sense embeddings, as proposed by Song et al. [17], allow for multiple senses by representing each sense instance as its own vector.

However, neither traditional word embeddings, like word2vec, nor sense embeddings consider the temporal aspect and assume that words are static across time. This creates a challenge in the face of dynamic and changing natural language [1]. Word embeddings can be used for temporal tasks if multiple embeddings are trained separately for separate time periods. As those embeddings are trained separately, they do not lie in the same semantic embedding space [10]. To ensure that they are in the same space and thus are comparable, they need to be aligned with each other [1]. To resolve such alignment issues, dynamic or diachronic word embeddings were introduced [10]. Kim et al. solve this by training their model on the earliest time periods first. Using the obtained weights as the initial state for the next training phase, they move through subsequent periods, allowing the embeddings to gain complexity and pick up new senses [9]. Yao et al. present another idea, called dynamic word embeddings, where align-

ment is enforced by simultaneously training the word embeddings for different time periods [21]. The disadvantage of their approach is that it addresses either only the temporal semantic shift or multi-sense aspect of words. In contrast, our approach takes both aspects into consideration.

Another method for extracting word senses is through the use of graph clustering or community detection. This method builds a graph based on co-occurrence features from a corpus. Such graphs represent the relation of words to each other, and allows for the extraction of sense clusters through graph clustering. Automatic word sense change detection based on curvature clustering can help understand the different senses in historic archives [18]. The proposed algorithm builds clusters of different senses. Their manual evaluation chose 23 terms with known sense changes (e.g., "gay"). Hope and Keller introduce the soft clustering algorithm MaxMax [7]. They identify word senses by transforming the co-occurrence graph around a given word into an unweighted, directed graph.

Mitra et al. present an approach that builds graphs based on distributional thesauri for separate time periods [14]. From those graphs they extract so-called "ego-networks". With the randomized graph-clustering algorithm Chinese Whispers, sense clusters are induced for specific words from their ego-network [2]. A similar and more recent approach by Ustalov et al., performs the clustering step with the meta-algorithm WATSET. This algorithm uses hard clustering algorithms, such as Louvain or Chinese Whispers, to perform a soft clustering [19]. Hard clustering algorithms assign nodes to exactly one cluster, whereas soft clustering produces a probability distribution of cluster assignments.

Besides embeddings and graph models, topic modeling has been applied to WSI as well. Lau et al. model word senses as topics of a word by application of latent Dirichlet allocation (LDA) as well as non-parametric hierarchical Dirichlet processes (HDP). They also applied their approach on the field of novelty sense detection by comparing induced senses of words of a diachronic corpus containing two time periods for a self developed dataset of 10 words [11].

Jatowt and Duh provide a framework for discovering and visualizing semantic changes w.r.t. individual words, word pairs and word sentiment. They use n-gram frequencies, positional information and Latent Semantic Analyses to construct word vectors for each decade of the Google Books n-gram dataset and COHA. To derive changes of word senses, they calculate the cosine similarity of vector representations of the same word at different time points. The authors show results of a case study for semantic changes of single words, inter-decade word similarity and contrasting word pairs in 16 experiments with mostly different words [8]

Our approach is similar to that of Mitra et al. [14], but differs in that we build a simpler co-occurrence graph and compare four different clustering algorithms for optimal performance. Additionally, our approach interprets the results of comparing sense clusters automatically to rank a word according to the likelihood of having gained a new sense over time. Furthermore, we use more fine-grained time-slices, to more precisely narrow in on the point in time where a new sense

emerges, whereas Mitra et al. merely oppose two timeslices at once. This point can be visualized by our drill-down, which shows the forest of sense trees built from ego-networks. This ability makes our approach valuable as an exploration tool in the field of historical and diachronic linguists, specifically for hypothesis testing.

3 A Forest of Word Senses

Our approach can be divided into the three parts visualized in Figure 1 using a fictitious example: (i) construction of a co-occurrence graph, (ii) extraction of word senses from a word's ego-network in the form of sense clusters, and (iii) matching them over time to create a forest of sense trees. We separate the corpus into n time-slices C_t, which are handled as sub-corpora. Similar to Mitra et al. [14], our analysis is limited to nouns. For each sub-corpus, we construct a filtered co-occurrence graph and extract sense clusters for each word. Finally, these clusters are connected across time-slices to form sense trees.

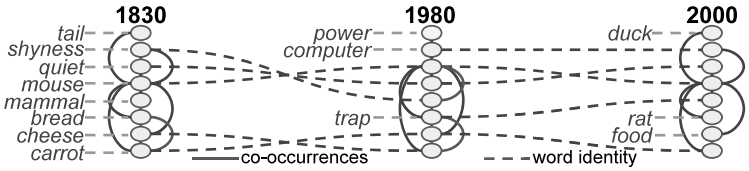
3.1 Building a Co-Occurrence Graph

To build a co-occurrence graph for time-slice C_t, we need to count how often words appear in the same context in C_t. Two nouns co-occur if they are part of the same sentence and the word-distance between them is $\leq n_{window}$. Therefore, the parameter n_{window} controls the complexity of the resulting co-occurrence graph. A smaller window size results in fewer co-occurrence edges and hence in a sparser graph. Having computed the co-occurrences, we can create the graph $G_t = (V_t, E_t)$ for each time-slice C_t. The sets of nodes and edges of G_t for time slice C_t are defined as

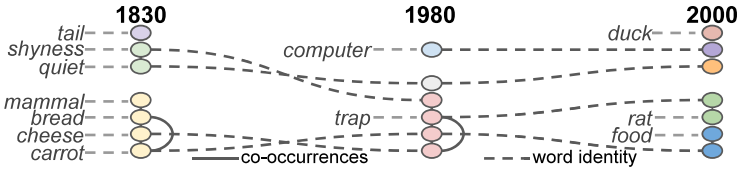
$$\begin{aligned} V_t &= \{u \in C_t \mid u \text{ is noun } \wedge \ tf(u) \geq \alpha_{tf}\} \\ E_t &= \{\{u, v\} \mid u, v \in V_t \ \wedge \ u \neq v \ \wedge \ cooc(u, v) \geq \alpha_{cooc}\} \end{aligned} \tag{1}$$

where $tf(u)$ is the term frequency of u in the given time-slice and $cooc(u, v)$ is the number of times words u and v appear within the same window. We use the threshold parameters α_{tf} and α_{cooc} to exclude rarely occurring words, as, depending on the window size n_{window}, the number of edges in this graph would rapidly increase. Therefore, our algorithm for detecting sense clusters operates on a filtered graph. Oftentimes, a corpus has an unbalanced distribution of data across all time-slices. As a result, some have much higher raw co-occurrence values. This especially affects frequently occurring words with generic meanings, such as *man* and *time*. Our initial filtering does not account for that. We use the point-wise mutual information (PMI) [20] $pmi(u, v) = cooc(u, v) / (tf(u) * tf(v))$ to reduce the importance of frequent words. After filtering more edges and possible unconnected nodes, the final co-occurence graph graph $G'_t = (V'_t, E'_t)$ for time-slice C_t, as depicted as one column in Figure 1a, is defined by

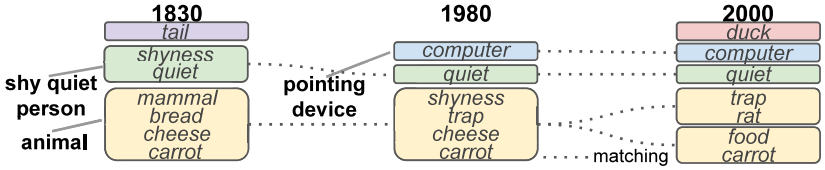
$$V'_t = \bigcup E'_t \qquad E'_t = \{\{u, v\} \in E_t \mid pmi(u, v) \geq \alpha_{pmi_t}\} \tag{2}$$



(a) Co-occurrence graph



(b) Ego-network for the word *mouse*



(c) Forest of sense trees

Fig. 1: Example for the construction of a simple sense forest of the word *mouse*

where α_{pmi} is a threshold parameter to remove the most loosely associated words in the given time-slice.

3.2 Word Sense Extraction

We use G'_t to extract information about word contexts in time-slice C_t. We hypothesize that the context of a word is an indicator for its senses as suggested by Lindén and hence use clustering to extract those senses [12]. The context of word w can be extracted in the form of an ego-network, which contains all neighbors of w and edges among them, but not w itself. Figure 1b shows such a network for the word *mouse*. The different colors indicate the different clusters of a time-slice. Formally, we define the ego-network $\widehat{G}_w = (\widehat{V}_w, \widehat{E}_w)$ of word w in time-slice C_t as

$$\begin{aligned} \widehat{V}_w &= \{u \in V'_t \mid \{u, w\} \in E'_t\} \\ \widehat{E}_w &= \{\{u, v\} \in E'_t \mid u \in \widehat{V}_w \ \wedge \ v \in \widehat{V}_w\} \end{aligned} \tag{3}$$

To extract different senses of w, we cluster the nodes of its ego-network $\widehat{G}_w$. In Section 4 we compare different clustering strategies. Each of these clustering algorithms produces a set of p disjoint clusters $S_{w_t} = \{c_1, \ldots, c_p\}$ from the ego-network of word w in time-slice C_t. Following Mitra et al. [14], we assume that

each of the resulting clusters represents a "sense cluster". Ideally, each meaning of w in C_t is represented by exactly one sense cluster. By relaxing this condition and allowing more than one sense cluster for each sense, we are able to get better and more fine-grained results by the clustering algorithms.

3.3 Matching Word Senses Over Time

In the last step, sense clusters are matched across time-slices. We use the Jaccard similarity to compare sets of words, which are given by the clusters of word w from two neighboring time-slices C_t and C_{t-1}. Let $c_i \in S_{w_t}$ be a sense cluster for word w in time-slice C_t. After a pairwise comparison between all sense clusters across two time-slices, we use a greedy approach to iteratively match those with the highest score. However, if there is no cluster $c'_j \in S_{w_{t-1}}$ that shares any words with $c_i \in S_{w_t}$, it remains unmatched. In this case, it would become the root cluster of a new sense tree.

A disadvantage of this matching strategy is that a word sense might simply not occur in a some time-slice and thus interrupt the lineage of that word sense. This may happen with sense clusters whose words have low frequencies, such as words that appear in specific scientific literature. Because they cannot always be matched between neighboring time-slices, we match them across a longer time span instead. More specifically, the sense clusters in S_{w_t} are matched not only to the sense clusters in $S_{w_{t-1}}$, but also to any other sense cluster in $S_{w_1}, \ldots, S_{w_{t-2}}$ that was not matched by a sense cluster in a later time-slice. We call this matching strategy leaf-matching.

$$\begin{aligned} S'_{w_{t-1}} = S_{w_{t-1}} \cup \bigcup_{i=1}^{t-2} \{c' \in S_{w_i} \mid c' \text{ not matched to any } c'' \in \bigcup_{j=i+1}^{t-1} S'_{w_j}\} \\ \arg\max_{c'} sim(c') = \{c' \in S'_{w_{t-1}} \mid sim(c', c_i)\} \end{aligned} \tag{4}$$

This comparison happens for each time-slice and produces a forest of sense trees $F_w = (V_w, E_w)$:

$$\begin{aligned} V_w &= \bigcup_{i=1}^{n} S_{w_i} \\ E_w &= \{(c, c') \mid c \in S_{w_i} \ \wedge \ c' \in S_{w_j} \ \wedge \ i < j \ \wedge \ c \text{ matched } c'\} \end{aligned} \tag{5}$$

F_w contains sense clusters without incoming edges. These clusters are root clusters and represent the beginning of a sense tree. Given a root cluster r, the respective sense tree $F_{w_r} = (V_{w_r}, E_{w_r})$ is defined as follows:

$$\begin{aligned} V_{w_r} &= \{c \in V_w \mid \exists p = ((r, c'_1), \ldots, (c'_k, c))\} \\ E_{w_r} &= \{(c, c') \in E_w \mid c, c' \in V_{w_r}\} \end{aligned} \tag{6}$$

Such a sense tree represents a distinct sense of the word w. For example, in Figure 1c, the matched clusters make up three sense trees, referring to three different meanings of the word *mouse*.

4 Comparison of Algorithms for Sense Clustering

In this section we introduce the following algorithms for the clustering step in Section 3.2: Chinese Whispers [2], Girvan-Newman [6], and Louvain [3]. We also describe the insights gained by setting up experiments using our drill-down to inspect the resulting clusters of these algorithms.

Chinese Whispers is an agglomerative clustering algorithm introduced by Biemann [2]. It does not have a fixed number of clusters, which is a property that suits word senses whose number is not known apriori. Its only parameter is the number of iterations. Depending on the size of the graph, a small number of iterations might never produce larger sense clusters. We chose 1 000 iterations to ensure that the algorithm converges as suggested by Biemann [2].

Inspecting the computed clusters, we discovered that Chinese Whispers produces one very large sense cluster that contains nearly every word in the ego-network, along with very few other small sense clusters. Since the large sense cluster contains more than one meaning, the results are not fitting our use case.

Girvan-Newman is a community detection algorithm named after its authors [6]. The algorithm is a hierarchical method that iteratively removes edges from the graph to find communities. It always removes the edge with the highest betweenness centrality or a custom metric, such as the co-occurrence frequency. Hence, the initially computed sense cluster contains all nodes of the ego-network. With each iteration, it produces more detailed sense clusters. This algorithm produces also a variable number of clusters and hence fits well for the task of identifying an unknown number of word senses. However, in our configuration, the computational costs of Girvan-Newman are around 25-30 times higher than those of Louvain and Chinese Whispers. Due to this high computational cost, we choose only three iterations. Because the fine-granularity of the hierarchy depends on the number of iterations, Girvan-Newman effectively produces one large cluster that contains most words of an ego-network in addition to many one-node sense clusters. As with Chinese Whispers, these results do not fit the use case of identifying multiple meanings of a word.

Louvain is a hierarchical community detection algorithm proposed by Blondel et al. [3]. It uses the Louvain modularity, which measures the difference of edge density inside communities to the edge density outside communities, to identify communities in a graph. Similarly to the previous algorithms, its number of detected clusters is not fixed. From the computed hierarchy it greedily chooses the graph partition that optimizes the algorithm's modularity measure. While investigating the computed sense clusters, we found that this partitioning produced a fairly balanced amount of sense clusters in terms of the cluster size. The results are applicable for our use case, since in most cases, sense clusters can be assigned to exactly one meaning of a word. However, in the later time-slices with an increasing number of documents and thus co-occurrences, the computed sense clusters are not partitioned as well. In some cases it produces quite large sense clusters that contain multiple meanings of a word.

5 Evaluation

The evaluation of WSI approaches is an open challenge, as there are no standardized experimental settings or gold standards. Kutozov et al. address the need for standardized, robust test sets of semantic shifts in their 2018 survey [10]. Other researchers created manually selected word lists, which can vary widely and make it difficult to compare the accuracy of approaches [14,21]. In this section, we introduce our evaluation data, which we compiled by aggregating the different approaches used in related work and annotated with the help of expert linguists. We also discuss the impact of the hyper-parameters of our approach and demonstrate the effectiveness qualitatively.

5.1 Compiled Word List for Evaluation

The word list compiled by Yarowsky [22] is used in several publications on word sense disambiguation [16]. Yarowsky developed an algorithm that disambiguates the following twelve words with two distinct meanings: *axes*, *bass*, *crane*, *drug*, *duty*, *motion*, *palm*, *plant*, *poach*, *sake*, *space*, and *tank*. We extended this list by adding new dictionary entries, novel technical terms and words based on related work [15]. It also contains words that did not gain new meanings in the last 200 years. For the lexicographical approach, the words were annotated with respect to new sense gain using word entries from the Oxford English and Merriam-Webster Dictionaries. WordNet[1], a commonly used lexical resource for computational linguistics, groups words with regard to word form as well as meaning. To merge the result obtained from the two different approaches we use a logical OR-operation. Thus, a word is considered to have gained a new sense if it is labeled positively from either of our two approaches.

In total we compiled a set of 112 words with either a single sense, multiple, but stable senses, or words that gained at least one new sense. Each candidate word was annotated by 15 linguists (C1 and C2 level) as *"Gained an additional sense since 1800"* or *"No additional sense since 1800"*. We measure an overall agreement of 61% and a free-marginal Fleiss kappa of 0.42 (fixed-marginal: 0.22). Based on a simple majority vote, 42 words gained an additional sense, whereas 70 did not. For some words, such as *android*, *beef*, or *pot*, we saw a high annotator agreement over 75%. When using this as a threshold, 14 words gained an additional sense, whereas 31 did not. The linguists where particularly undecided on the words *cat*, *honey*, *power*, and *state*.

5.2 Corpus of Historical American English (COHA)

For demonstrating our proposed approach, we use the Corpus of Historical American English (COHA)[2]. It is one of the largest temporal corpora over one of the

[1] `https://wordnet.princeton.edu/`
[2] `https://www.english-corpora.org/coha/`

Table 1: Comparison of graph clustering algorithms

Algorithm	Precision@5	Precision@10	Precision@20
Chinese Whispers	0.4	0.4	0.35
Girvan-Newman	0.2	0.5	**0.55**
Louvain	**0.6**	**0.6**	0.4

longest time ranges [4]. COHA spans texts from the years 1810 to 2009 and contains more than 100,000 single texts in fiction, popular magazines, newspapers and non-fiction books with a total of 400 million words.

We split the corpus by decade to generate the time slices, since vastly different vocabularies would result in different word contexts. Ideally, the vocabulary and word distribution is relatively stable across all time-slices. Since the number of tokens increases with each decade, we measure the vocabulary overlap. In our measurements (not shown due space constraints), neighboring decades share 20-30% of their vocabulary, while the decades that are further away from each other share only 5-15% of their vocabulary for both measures. Very high values produced by the cosine similarity highlight that frequently occurring words appear in most decades. We can conclude that using a frequency- and co-occurrence-based approach to extract information about changes of word senses over time is feasible. For a deeper statistical analysis of the corpus, we refer readers to the work by Jatowt et al. [8]

5.3 Hyper-Parameter Evaluation

We evaluate our approach using the precision for the top k ranked words. We use the values 5, 10, and 20 for k. The ranking is done with a score based on sense trees introduced in Section 3. This scoring measure counts the number of sense trees produced for a word. It only counts sense trees representing new senses, i.e., starting in the second time-slice or later. sense trees with a length of 1 are ignored. With this ranking, we expect words that gained new senses to appear at the top of the ranking and words that did not gain a new sense to appear at the bottom.

The following parameter settings were found by optimizing our approach on the annotated word list. We set the pointwise mutual information (PMI) threshold parameter $\alpha_{pmi} = 0.01$, we used the Jaccard similarity as similarity measure, and we used leaf-matching to match sense clusters across time-slices.

We compare the three graph clustering algorithms introduced in Section 3: Chinese Whispers, Girvan-Newman, and Louvain. Table 1 presents the results with these algorithms. The highest precision values are highlighted bold.

Louvain outperforms the other two algorithms at $k = 5$ and $k = 10$. Girvan-Newman has a much lower precision at $k = 5$, but performs much better at the values 10 and 20 for k. Chinese Whispers does not perform as well. We suggest Louvain as clustering algorithm, because it fits best for our use case that a sense cluster should only represent a single meaning.

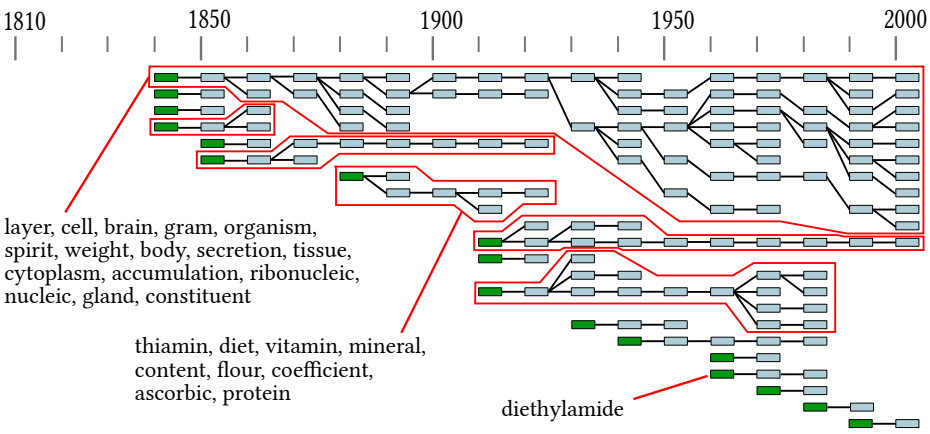


Fig. 2: Sense tree for the word *acid*. Green boxes represent the initial cluster for a sense, matched clusters are connected by straight lines.

5.4 Qualitative Evaluation

To qualitatively evaluate our approach, we take a closer look at the drill-down example for the word *monitor*. Our approach produced five sense trees, each of which can be matched to a single meaning of the word: *hall monitor* starting in the 1830s, the warship of that name starting in the 1860s (two sensetrees), *monitor* in the technological sense of a screen starting in the 1920s, and the surveillance sense starting in the 1960s. The time periods at which new senses emerged are accurate. For example, the warship was created in the 1860s and that is also the time-slice in which we detect that sense.

One weakness of the approach is that although the earliest sense tree can be interpreted easily as sense of *hall monitor*, a closer look reveals that the clusters are only matched by the word *master*. This shows that the matching can be influenced easily by just a few words. Consequently, the sense of a single sense tree can easily drift apart.

Another weakness is that in fact two sense trees represent the meaning of warship. In this case, the clustering algorithm is too strict, which results in different clusters for the same sense in the first time-slice of those sense trees.

The later sense trees represent distinct senses of *monitor*: the technological sense and the surveillance sense. However, the branches of these sense trees are not separated sufficiently. For example, *monitor* gained the sense as *TV screen* only through the word *press*, which co-occurs in one cluster with *science*. This downside is related to the sense drift mentioned above. The consequence is that sense clusters for new senses match existing sense trees instead of becoming a new sense tree.

Our graph representation of a text corpus can be used to visually explore linguistic features. Figure 2 shows the forest of sense trees for the word *acid*. Each of the boxes represents a set of words $c_i \in S_{w_t}$ as defined before. Each

sense tree begins with a green box, the following clusters that were matched across time-slices are connected by straight lines. While some sense trees only appear for a few decades, others branch out or persist in a single branch for longer periods. Interestingly, we can see the discovery of DNA in the late 19th century and LSD in the 1960s. We developed an early prototype that we used to explore word clouds for each sub-cluster or matched component across time.

6 Limitations and Future Work

Although our approach is able to identify different senses of some words, there is still room for improvement: graph clustering algorithms and matching strategies, the evaluation of different window sizes used during the co-occurrence graph creation, and the conduction of a survey to obtain a word dataset that can be used as gold standard for evaluating temporal WSI approaches. Useful features make our approach available as an explorative tool, adjustments in the used language models to make our approach applicable to historic language, a quantitative comparison to word embeddings, verification of the stability of our approach, a custom slicing of time periods, sensitivity for differently spelled variants of the same word, and detecting not only the birth of new word senses but also the death of word senses. However, our approach struggles to create well-partitioned sense clusters for different senses, which also affects the matching strategies. Additionally, these strategies do not prevent sense drifting and sense trees may change their meaning significantly over multiple time-slices.

References

1. Bamler, R., Mandt, S.: Dynamic word embeddings. In: Proceedings of the International Conference on Machine Learning (ICML). pp. 380–389. JMLR Inc. and Microtome Publishing (2017)
2. Biemann, C.: Chinese whispers: an efficient graph clustering algorithm and its application to natural language processing problems. In: Proceedings of the Workshop on Graph-based Methods for Natural Language Processing (TextGraphs). pp. 73–80. Association for Computational Linguistics (2006)
3. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory and Experiment **2008**(10), P10008 (2008)
4. Davies, M.: Expanding horizons in historical linguistics with the 400-million word corpus of historical american english. Corpora **7**(2), 121–157 (2012)
5. Feuerbach, T., Riedl, M., Biemann, C.: Distributional semantics for resolving bridging mentions. In: Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP). pp. 192–199. Association for Computational Linguistics, Stroudsburg, PA, USA (2015)
6. Girvan, M., Newman, M.E.: Community structure in social and biological networks. Proceedings of the national academy of sciences **99**(12), 7821–7826 (2002)
7. Hope, D., Keller, B.: MaxMax: A graph-based soft clustering algorithm applied to word sense induction. In: Proceedings of the International Conference on Computational Linguistics and Intelligent Text Processing (CICLing). pp. 368–381. Springer-Verlag, Heidelberg, Germany (2013)

8. Jatowt, A., Duh, K.: A framework for analyzing semantic change of words across time. In: Proceedings of the IEEE/ACM Joint Conference on Digital Libraries (JCDL). pp. 229–238 (2014)
9. Kim, Y., Chiu, Y.I., Hanaki, K., Hegde, D., Petrov, S.: Temporal analysis of language through neural language models. In: Proceedings of the Workshop on Language Technologies and Computational Social Science@ACL. pp. 61–65. Association for Computational Linguistics (2014)
10. Kutuzov, A., Øvrelid, L., Szymanski, T., Velldal, E.: Diachronic word embeddings and semantic shifts: a survey. In: Proceedings of the International Conference on Computational Linguistics (COLING). pp. 1384–1397. Association for Computational Linguistics (2018)
11. Lau, J.H., Cook, P., McCarthy, D., Newman, D., Baldwin, T.: Word sense induction for novel sense detection. In: Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL). p. 591–601. Association for Computational Linguistics (2012)
12. Lindén, K.: Evaluation of linguistic features for word sense disambiguation with self-organized document maps. Computers and the Humanities **38**, 417–435 (2004)
13. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. In: Proceedings of the International Conference on Learning Representations (ICLR). pp. 1–12 (2013)
14. Mitra, S., Mitra, R., Maity, S.K., Riedl, M., Biemann, C., Pawan, G., Mukherjee, A.: An automatic approach to identify word sense changes in text media across timescales. Natural Language Engineering **21**(5), 773–798 (2015)
15. Navigli, R.: Word sense disambiguation: A survey. ACM Computing Surveys **41**(2), 1–69 (2009)
16. Rapp, R.: Word sense discovery based on sense descriptor dissimilarity. In: Proceedings of Machine Translation Summit (MTSummit). pp. 315–322. European Association for Machine Translation (2003)
17. Song, L., Wang, Z., Mi, H., Gildea, D.: Sense embedding learning for word sense induction. In: Proceedings of the Joint Conference on Lexical and Computational Semantics (*SEM). The *SEM Organizing Committee (2016)
18. Tahmasebi, N., Risse, T.: On the uses of word sense change for research in the digital humanities. In: Proceedings of the International Conference on Theory and Practice of Digital Libraries (TPDL). pp. 246–257. Springer-Verlag (2017)
19. Ustalov, D., Panchenko, A., Biemann, C., Ponzetto, S.P.: Watset: Local-global graph clustering with applications in sense and frame induction. Computational Linguistics **45**(3), 423–479 (2019)
20. Yang, H., Callan, J.: A metric-based framework for automatic taxonomy induction. In: Proceedings of the International Joint Conference on Natural Language Processing (IJCNLP). pp. 271–279. Association for Computational Linguistics (2009)
21. Yao, Z., Sun, Y., Ding, W., Rao, N., Xiong, H.: Dynamic word embeddings for evolving semantic discovery. In: Proceedings of the ACM International Conference on Web Search and Data Mining (WSDM). pp. 673–681. ACM Press (2018)
22. Yarowsky, D.: Unsupervised word sense disambiguation rivaling supervised methods. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL). pp. 189–196. Association for Computational Linguistics (1995)