An Investigation of Sense Disambiguation in Scientific Texts

Manav Rathod

Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2022-132
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2022/EECS-2022-132.html

May 16, 2022

Acknowledgement

Thank you to Marti Hearst and Doug Downey for their advice and support on this project throughout this past year.

I would also like to thank the members of the Hearst Lab group for their comments on this work. Special thanks to Katie Stasaski for her years of wonderful mentorship.

This work was supported in part by a gift from the Allen Institute for AI.

Lastly, thank you to my friends and family.

An Investigation of Sense Disambiguation in Scientific Texts

by Manav Rathod

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, in partial satisfaction of the requirements for the degree of **Master of Science, Plan II**.

Approval for the Report and Comprehensive Examination:

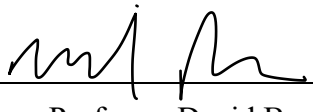
Committee:

Professor Marti Hearst
Research Advisor

5/13/22

(Date)

* * * * * * *



Professor David Bamman
Second Reader

5/13/22

(Date)

An Investigation of Sense Disambiguation in Scientific Texts

Manav Rathod
UC Berkeley
manav.rathod@berkeley.edu

Abstract

Word Sense Disambiguation (WSD) is a well-researched problem in Natural Language Processing with decades of papers written about it and new techniques coming out every year. However, this technique is still under-explored in certain domains such as scientific texts. Scientific papers are a particularly interesting use case as there is a rich amount of information associated with each paper that can potentially improve existing approaches for disambiguation, mainly its title, abstract, and its place in the citation graph. A related area of study that also works in this direction is Acronym Disambiguation (AD). We believe that these two problems are related and similar techniques could strongly perform in both settings. However, there is a lack of a large dataset for WSD in scientific texts, motivating the need to create one artificially, without spending an exorbitant amount of resources. Thus, we turn towards Pseudowords as a means of creating this dataset. We demonstrate that using paper information can lead to improvements in AD and WSD and present a brand-new dataset to further research in Scientific WSD.

1 Introduction

Word Sense Disambiguation (WSD) is a long-studied problem in natural language processing. A central theme across many of the methods that approach this task is to compare the context a word is used in practice against the typical contexts a particular sense of the word is used. Since the rise of distributed representations of words as vectors, the comparison has now turned towards comparing vector representations of words against representations of senses (Tyagi et al., 2022). Now, in the era of transformers (Vaswani et al., 2017) and pre-trained models, these representations have mainly been contextual representations learned via BERT-like models (Devlin et al., 2019).

Recent work has focused on methods for improving the quality of embeddings produced by neural language models, often relying on existing corpora such as WordNet as a knowledge base to enhance representations (Loureiro et al., 2022) or finding ways to better incorporate gloss information (Huang et al., 2019; Blevins and Zettlemoyer, 2020; Bevilacqua et al., 2020; Barba et al., 2021). This reliance on a knowledge-base has been shown to be useful, but does not help in the context of the general scientific domain where there does not exist an all encompassing well-defined ontology of terms and where new terms are often being introduced. These other gloss-based methods do not demonstrate performance on scientific tasks.

In fact there has been little recent work on WSD in the domain of scientific text besides work mainly in the biomedical literature (Duque et al., 2018). Furthermore, methods also tend to rely on an ontology of concepts (Prokofyev et al., 2013), which do not exist for other scientific fields.

A likely reason that scientific WSD has seldom been explored beyond the biomedical domain is the lack of datasets to address this niche. Since the introduction of the WSD task, significant resources have been dedicated to creating large, rich datasets to sufficiently train models on English WSD. A typical suite used for evaluation was created by Raganato et al. (2017), which contains the SemCor dataset (Miller et al., 1994) for training and various SemEval (Pradhan et al., 2007; Navigli et al., 2013; Moro and Navigli, 2015) and SenseEval (Edmonds and Cotton, 2001; Snyder and Palmer, 2004) datasets for validation and testing.

Yet, researchers have found ways around limited datasets by creating artificial datasets using pseudowords. Pseudowords are a technique for creating artificial datasets for word sense disambiguation tasks by creating fake ambiguous words by concatenating unambiguous words into a single word and then replacing instances of each word with the pseudoword. e.g. we replace sentences that contain banana and door with banana-door. First

introduced in Gale et al. (1992b), pseudowords were originally constructed by randomly selecting unambiguous words. Subsequent work found ways to improve the quality of pseudowords by picking senses that more closely matched the sense distributions found in real ambiguous words.

However, the rise of large-scale datasets suitable for pre-trained models seems to have lessened the need for these artificial datasets for English WSD and research in this field has faded. Yet, there are niche domains, e.g., scientific papers, and many languages that do not have access to similar large datasets (Lu et al., 2006; Kim and chul Kwon, 2021), so we believe there is still potential in exploring methods to revive pseudowords to create artificial datasets.

We are particularly interested in scientific documents given the amount of the time researchers spend reading papers (Tenopir and King, 2008) and the known difficulty of the process (Bazerman, 1985). There are many terms such as "kernel" and "transformer" that have very different meanings depending on the scientific field, so there is a unique need to build models to help resolve this ambiguity. Scientific papers are also unique as there is lots of additional information derived from knowing a paper is a research paper. Mainly, there is a title, abstract, and also a whole set of papers that the paper cites and also lots of papers that cite the paper. All together, these papers form part of the overall citation graph. The place of a paper within the citation graph might give hints into the true meaning of ambiguous words. Current WSD methods never explore how to use this additional information.

Another related area to WSD is the task of Acronym Disambiguation (AD). The task is almost the same as WSD except instead of having polysemous words we have acronyms with multiple potential expansions e.g., CNN to Convolution Neural Networks or Cable News Network. There has been particular recent interest in AD for scientific text with the introduction of the SciAD dataset (Pouran Ben Veyseh et al., 2020), a large dataset of sentences containing acronyms with multiple potential expansions. The creators of this dataset ran a shared task to encourage researchers to study this problem, however, the top performers still fell short of human performance (Veyseh et al., 2021). Additionally, this dataset is quite limited as there was not any particular system in place for dividing terms between the train, development, and test sets and there are lots of noisy labels. Furthermore, it does not enable models to use information about the paper each example sentence comes from, since paper information is not provided.

Since these WSD and AD are related we believe that similar approaches can be developed for each task. Thus, the main contributions of this work can be summarized as follows:

- We present a novel method for scientific acronym disambiguation that outperforms the current state of the art, reaching human level performance.
- We resurrect pseudowords using modern semantic similarity techniques and re-establish its relevance for creating useful datasets for WSD.
- We show that our method for acronym disambiguation also performs well on our artificial pseudowords dataset for scientific WSD.

2 Related Works

2.1 Word Sense Disambiguation

WSD methods now mainly leverage pretrained-transformers like BERT (Devlin et al., 2019) and develop clever mechanisms for best utilizing context and gloss information, while enabling the model to be robust to unseen or rare senses. GlossBERT (Huang et al., 2019) takes a simple approach and just concatenates each target sentence with all possible glosses for a word and passes it into a BERT model which outputs a representation which is then used as an input into a classification layer. BEM (Blevins and Zettlemoyer, 2020) breaks this process into two steps by training two different BERT-based encoders, one for the sentence and one for the gloss. Escher (Barba et al., 2021) outperforms both of these methods by moving beyond these embedding-based scoring approaches and instead reframes WSD as Extractive Sense Comprehension. In this task, a model picks a span of sense text as being most align with the terms use in a sentence. All of these models worked to improve the state of the art in English WSD, but have not been put to use in Scientific WSD.

2.1.1 Pseudowords

Shortly after the introduction of pseudowords, it was shown that these artificial datasets were not comparable in difficulty to real WSD datasets

(Gaustad, 2001). Thus, future works attempted to construct more realistic pseudowords by picking pseudosenses that have more semantically similar senses and follow distributions seen in real ambiguous words. Nakov and Hearst (2003) selected pseudowords using lexical category information from MeSH (Lipscomb, 2000). Otrusina and Smrz (2010) developed similarity based methods using WordNet (Fellbaum, 2000) to find semantically related terms and by using TF-IDF scoring. Pilehvar and Navigli (2014) creates one of the largest pseudoword datasets using WordNet to find semantically related words by generating topic signatures. Work in this area has stalled significantly in recent years as large-scale WSD resources have grown, reducing the need for artificial datasets. However, as new areas such as the general scientific domain opens itself to WSD, there is potential to revitalize this method using modern techniques.

2.2 Acronym Disambiguation

The techniques within this space are similar to WSD methods and focus primarily on neural methods (Pan et al., 2021) and techniques to create better representations (Pouran Ben Veyseh et al., 2021). The current state of art on the SciAD dataset, DeepBlueAI, (Pan et al., 2021) takes an approach similar to GlossBERT and simply concatenates each possible acronym expansion with the target sentence to generate an output from a SciBERT (Beltagy et al., 2019) model which is then passed into a classification layer. While this method did quite well on the shared task that was run, it is still shy of human-level performance. Furthermore, it does not use any intrinsic properties of the paper each target sentence came from since the current version of the SciAD dataset does not provide it.

2.3 Citation-Informed Methods

Often when working on tasks in the scientific domain, it is useful to use the additional information that comes with knowing a document is a published paper, for example, its place in the citation graph. Various methods have used citation information for several different tasks. Caragea et al., 2014 uses citation-based features to train a Naive-Bayes classifier for keyword detection. Garg et al., 2021 also used citation information for keyphrase generation by taking into consideration of sentences in cited papers that form a good summary. Viswanathan et al., 2021 used citation information to improve performance on Scientific IE tasks by making citances (Nakov et al., 2004) features for relation extraction and salient entity classification. Lastly, Cohan et al., 2020 learned embeddings for papers via a contrastive learning objective that forced papers that cited each other to have similar embeddings. These citation-informed embeddings improved performance in a variety of scientific tasks like citation prediction and document classification. However, to the best of our knowledge, citation information has not yet been used for WSD/AD.

3 Scientific Acronym Disambiguation

3.1 Task

In AD, we are a given an input text $\mathbf{x}$ which is a sequence of n words: $\mathbf{x} = w_1, \ldots, w_i, \ldots, w_n$, where w_i represents an acronym. For each acronym, we also have a list of k possible expansions $e_1, \ldots, e_k$, where e_j is the correct intended expansions of w_i. The goal of this task is to select the correct expansion for each input.

Given we are specifically looking at disambiguating acronyms appearing in sentences in scientific papers, we extend the amount of information given in this task by also including the paper title and abstract, $\mathbf{p}$, for each example.

An example of an input is shown in Figure 1.

Title: A Taxonomy of Deep Convolutional Neural Nets for Computer Vision
Abstract: Traditional architectures for solving computer vision problems and the degree of success ...
Text: We start with " AlexNet " as our base **CNN** and then examine the broad variations proposed over time to suit different applications.
Possible Expansions:

- Convolutional Neural Network
- Condensed Nearest Neighbor
- Complicated Neural Networks
- Citation Nearest Neighbour

Output: Convolutional Neural Network

Figure 1: Example of Acronym Disambiguation.

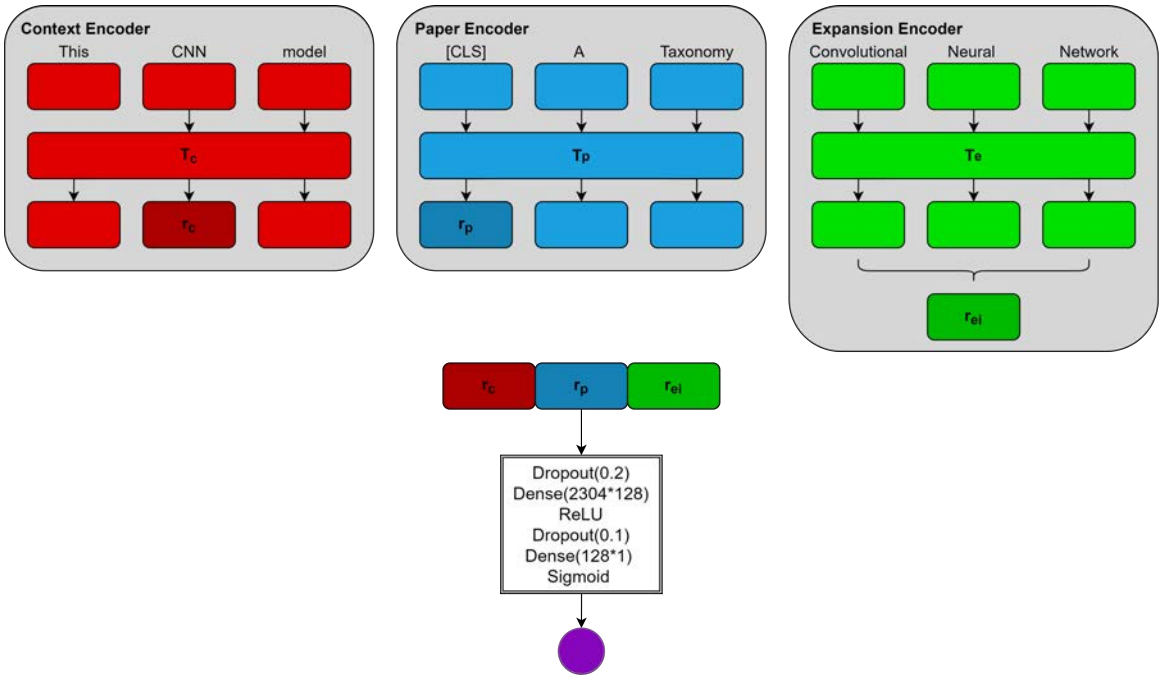


Figure 2: STARDUST model for acronym disambiguation.

3.2 Methodology

In this section, we propose our approach for **S**cientific **T**ext **AcR**onym **D**isambiguation **US**ing **T**riEncoders, STARDUST.

STARDUST combines ideas from DeepBlueAI Pan et al. (2021) and BEM Blevins and Zettlemoyer (2020). The overall architecture is visualized in Figure 2. We treat disambiguation as a binary classification task and independently determine for each expansion the probability that it fits in a given input. To do this, our model consists of three encoders and a classifier head.

Our three encoders are (1) a context encoder, which represents the target acronym and its surrounding context, (2) a paper encoder, which represents the paper a target sentence is from, and (3) an expansion encoder that embeds each expansion for an acronym.

For the context and expansions encoders, we use a pre-trained Sentence-BERT (Reimers and Gurevych, 2019) model[1] found in the HuggingFace transformers model hub (Wolf et al., 2020). We decided on using a Sentence-BERT model since it has been shown that these models produce much stronger representations compared to original BERT models. For the paper encoder, we use SPECTER (Cohan et al., 2020) (also using the transformer library) since we believe papers that cite each other will use acronyms in similar ways. Thus, having similar embeddings for papers that cite each other will be useful.

For the classifier head, we use a two-layer feed-forward network with a ReLU activation for the hidden layer, a sigmoid activation for the output layer, and dropout (Srivastava et al., 2014) between each layer.

To calculate this probability, we use our three encoders to embed the input context, paper title+abstract, and expansion. Then, we concatenate these three embeddings and pass it into our classifier head, which outputs a probability.

More precisely, the context encoder, which we will call T_c, takes as input a sentence $\mathbf{x}$ containing an acronym w_i. This encoder outputs a sequence of representations

$$\mathbf{C} = T_c(\mathbf{x}).$$

We are specifically interested in

$$r_c = \mathbf{C}[i],$$

which is the embedding corresponding to the acronym whose expansion we want to predict. If the acronym is broken up into multiple subwords by the tokenizer, we represent the word by the average of the subword embeddings. For example, if

[1] https://huggingface.co/sentence-transformers/all-mpnet-base-v2

the acronym w_i is broken up into the l-th to k-th tokens, then

$$r_c = \frac{1}{k-l}\sum_{j=l}^{k} \mathbf{C}[j].$$

Next, the paper encoder, which we will call T_p, takes as input a paper's title and abstract, $\mathbf{p}$, padded with the '[CLS]' token at the beginning and the '[SEP]' token at the end, This also outputs a sequence of representations, $\mathbf{P}$. Following the SPECTER paper, we produce an embedding by simply taking the embedding vector corresponding to the '[CLS]' token:

$$r_p = \mathbf{P}[0].$$

The expansion encoder, which we will call T_e, takes as input an acronym expansion e_i and outputs a sequence of representations for all the l subwords in e_i. We then take an average over all of them to get

$$r_{e_i} = \frac{1}{l}\sum_{j=0}^{l} T_e(e_i)[j].$$

Now, we concatenate our three representations input a single vector:

$$z_i = [r_c; r_p; r_{e_i}].$$

We pass z_i into the classifier head, H, which outputs a probability:

$$s_i = H(z_i).$$

We compute probabilities $s_1, \ldots, s_k$ for all k possible expansions for the acronym w_i. During evaluation, we predict the expansion with the highest probability.

To train our model, we use a binary cross-entropy loss on the probabilities for the expansions. So, the loss for an expansion probability s_i is

$$L(s_i) = \begin{cases} -\log s_i, & i = j \\ -\log(1 - s_i), & i \neq j \end{cases}$$

where s_j is the probability corresponding to the correct expansion e_j.

To efficiently train this architecture, we freeze the pre-trained models and only update our feedforward layer to learn the best mechanism for scoring.

We train our model using a learning rate of 1e-3 using the Adam optimizer (Kingma and Ba, 2015) for 10 epochs with a batch size of 1. Training takes about 2 hours on a NVIDIA Titan RTX.

3.2.1 Context Enhancement

To increase the amount of context for an acronym, we also search for an additional example of the target acronym w_i being used elsewhere in the paper, selecting the first found example, $\mathbf{y}$. We then update our context representation, r_c, to be the average of the embeddings for these two sentences:

$$r_c = \frac{T_c(\mathbf{x})[i] + T_c(\mathbf{y})[i]}{2}$$

If no additional examples are found, we just use the representation defined previously.

3.2.2 Using the Citation Graph

The paper embedding is a good indicator about the field a paper belongs to (Cohan et al., 2020). One reason for this might be that SPECTER is trained to produce similar embeddings for papers that cite each other and papers that cite each other likely belong to the same field. Our key insight into choosing to use SPECTER is that it can bring this understanding of a paper's field using citation information since field information is likely useful in disambiguating acronyms.

However, this is still an indirect way of representing citation information and it can potentially be enhanced. So, we try out the following to create a more direct citation-informed representation of the field. We look at the set of all papers in the S2ORC dataset that are cited by or cite the paper the input text is from. We denote this set as C. For convenience, we will also include the original paper in C. Then we use SPECTER to create embeddings for all the title+abstract's of these papers, $\mathbf{p_c}$, and average them into a single representation. Thus,

$$r_p = \frac{1}{|C|}\sum_{c \in C} T_p(\mathbf{p_c})[0]$$

The intention is that averaging over the citation neighborhood would emphasize the commonalities among the papers in the citation neighborhood and allow the model to be more robust.

3.3 Evaluation

To evaluate our approach, we train and test our model on the SciAD dataset. However, this dataset does not natively contain the paper information we need to train our model. To remedy this we downloaded the S2ORC dataset (Lo et al., 2020) and

built an index over it using ElasticSearch[2]. We then searched for each input sentence from the SciAD dataset in the index using an exact match query and retrieved the paper information. However, we were not able to match all examples to papers, potentially due to formatting errors or papers not being present in the S2ORC dataset. In the end, we were able to recover papers for 20911 out of 50034 training examples and 2589 out of 6189 development set examples from SciAD. We do not use the test set because the labels for it were not made public, so all results shown will be for the development set.

3.4 Results

We wished to compare against the current state of the art given by (Pan et al., 2021) (DeepBlueAI). Unfortunately, comparing to the numbers given in the paper is not an exact comparison as their method used all of the development set, while we were only able to use a fraction of due to the limitations described earlier. To thoroughly evaluate our models and the impact of each of our design choices, we train 5 versions of our model. We show the results of our models' performances in Table 1.

First, we establish a simple baseline (DotProduct) that simply uses a dot product between the expansion embedding and the context embedding

$$s_i = r_c \cdot r_{e_i},$$

and predicts the sense with the highest score.

Next, we evaluate a model that only uses the context embedding and the expansion embedding with a feedforward network for scoring. We call this model BiEncoder. We also evaluate a version of this model using the context enhancement strategy described in 3.2.1. We call this model BiEncoder$_{\text{ce}}$

Similarly, we train three versions of the full STARDUST model. First, we have our base version (STARDUST). Then, we also utilize context enhancement in STARDUST$_{\text{ce}}$. Lastly, STARDUST$_{\text{ce+cg}}$ additionally uses an average of the paper embeddings of papers in the citation neighborhood as described in Section 3.2.2.

As we can see from the results, our best performing model gives us a 9 point improvement in F1 compared to our base version. The ablations we perform show that our various modifications over a simple baseline are valuable.

[2] https://github.com/elastic/elasticsearch

Model	Precision	Recall	F1
DeepBlueAI	0.9629	0.9106	0.9360
DotProduct	0.9040	0.5897	0.7138
BiEncoder	0.9387	0.7992	0.8634
BiEncoder$_{\text{ce}}$	0.9751	0.9002	0.9362
STARDUST	0.9780	0.9053	0.9403
STARDUST$_{\text{ce}}$	**0.9811**	**0.9225**	**0.9509**
STARDUST$_{\text{ce+cg}}$	0.9802	0.9123	0.9450

Table 1: Results of experiments on the SciAD development set. Note that DeepBlueAI was evaluated on a larger amount of data in the development set.

Context Enhancement is specifically very useful as the performance gains of BiEncoder$_{\text{ce}}$ vs. BiEncoder and STARDUST$_{\text{ce}}$ vs STARDUST are quite large (7.3 and 1 F1 points, respectively). This reflects the "one sense per discourse" results of Gale et al. (1992a).

We also see modest gains in using the Paper information as shown in the performance boosts of STARDUST over BiEncoder an STARDUST$_{\text{ce}}$ over BiEncoder$_{\text{ce}}$ (7.7 and 1.4 F1 points, respectively).

Interestingly, our method for paper enhancement using the citation graph to enhance the paper representation does not help the model and actually slightly worsens performance, but only by 0.5 F1 points. A likely explanation for this is that a lot of papers that are cited by a paper or cite a paper don't necessarily relate too strongly and may only share tangential details. Thus, the simple averaging over the citation neighborhood introduces a lot of noise in many cases in a way that decrease the strength of the representation. Further work is needed to discover techniques that more intelligently represent a paper using its citation neighborhood.

Though our results are not directly comparable with DeepBlueAI, they are promising. First, we are demonstrating strong performance despite having less training data. Second, DeepBlueAI's training procedure involves finetuning a SciBERT model, in addition to a feedforward classifier, while our methodology freezes the encoders, making it more efficient. Lastly, DeepBlueAI relied on complex training procedures such as Dynamic Negative Sample Selection, Task Adaptive Pretraining (Gururangan et al., 2020), Adversarial Training (Miyato et al., 2017), and Pseudolabeling (Iscen et al., 2019), while we were able to demonstrate strong performance using a simpler supervised training

objective.

3.4.1 Error Analysis

Next, we perform an error analysis to understand the types of mistakes our top-performing model makes. There are several examples that the model gets wrong, which we believe to be due to noisiness within the dataset. Examining the examples in Table 2, we can clearly see that the labels are incorrect. Just by looking at the title we can determine what the expansion should be. This is a main motivation for wanting to use title/abstract information in our model.

Text: Hence, **RS** proves to be robust since the rate does not saturate, and it is even more preferable at high SNR. **Prediction:** rate splitting **Correct Label:** relay station **Paper Title:** Rate-Splitting Robustness in Multi-Pair Massive MIMO Relay Systems
Text: We generate suitable misclassification costs for **DBN** using the training set. **Prediction:** deep belief network **Correct Label:** directed belief net **Paper Title:** A Cost-Sensitive Deep Belief Network for Imbalanced Classification
Text: In fact, the **ARD** model is nearly as robust as its teacher. **Prediction:** adversarially robust distillation **Correct Label:** accelerated robust distillation **Paper Title:** Adversarially Robust Distillation

Table 2: Examples that demonstrate noise in the SciAD dataset.

However, there are still several genuine errors made by our model. We can look at these in Figure 3. These require a keen understanding of the field and the context to determine the correct expansions. For example, a machine learning scientist would likely be able to connect "variational autoencoder" to "reparameterization trick" since it is a technique used in training them. However, it seems like the model is quite confused as "retweets" is not at all closely related. Further research is required to understand how we can design models that can infer these relationships given limited context.

3.4.2 Generalization

Lastly, we want to test the ability of our model to generalize to new expansions that may arise and can be added to the dictionary, but are not part of the training data. To do this, we specifically look at examples in the development set that are not part of the training set. This was not an intentional part of the dataset design, however, so the number of samples that fit this criteria are limited. There are 57 examples in our version of the development set that we can look at, which encompass 54 unique expansions. Our best trained model only achieves a 42.6% accuracy on these examples, which is significantly below performance on the overall development set. A sample of examples of that the model failed to generalize to can be found in Table 4.

Text: This solution enables the **RT** to be used with a variety of continuous distributions, including the Dirichlet distribution. **Prediction:** retweets **Correct Label:** reparameterization trick **Paper Title:** Nested variational autoencoder for topic modelling on microtexts with word vectors
Text: The original **RM** presented drawbacks that have been previously analysed. **Prediction:** resource management **Correct Label:** roofline model **Paper Title:** New Thread Migration Strategies for NUMA Systems
Text: DAGs in an **EC** are Markov equivalent; that is, given an observed dataset, they are statistically indistinguishable and represent the same set of independence assertions. **Prediction:** eigenvector centrality **Correct Label:** equivalence class **Paper Title:** Bayesian Structure Learning by Recursive Bootstrap

Table 3: Example in the SciAD dataset our model got wrong.

These examples should be able to be captured by a model that is capable of understanding the sentence context and the field that is being discussed. Further work is needed to developing a model that is more robust to new expansions and even new acronyms, which is unexplored in this dataset. The first step towards answering these questions would be a dataset that is designed with capturing this ability.

4 Scientific WSD

Now that we have demonstrated performance improvements on the AD task, we turn our attention to Scientific WSD.

Text: and references therein discuss simplification techniques in **CAS** systems further. **Prediction:** consensus attention sum **Correct Label:** computer algebra systems **Paper Title:** Simplifying Probabilistic Expressions in Causal Inference
Text: In the context of **ROM** and PDE, this decomposion is also know as Proper Orthogonal Decomposition (POD). **Prediction:** range of motion **Correct Label:** reduced-order models **Paper Title:** Appraisal of data-driven and mechanistic emulators of nonlinear simulators: The case of hydrodynamic urban drainage models
Text: Moving on, we execute the second step of the **MI** and provide the following theorem. **Prediction:** mutual information **Correct Label:** mathematical induction **Paper Title:** Robust Fuzzy-Learning For Partially Overlapping Channels Allocation In UAV Communication Networks

Table 4: Examples in the SciAD dataset that have expansions that are not in the training set.

4.1 Task

The setup for this task is identical to AD since instead of disambiguation an acronym with k possible expansions, we have k unique sense definitions for a given ambiguous word.

An example for this is the word term "transformer". To many researchers in the NLP community, a transformer (Vaswani et al., 2017) is a deep learning architecture built around the self-attention mechanism. However, to electrical engineers a transformer transfers electrical energy between devices. Our goal is to see if our AD model can also be trained to perform scientific WSD.

However, as noted in the introduction, there are not existing datasets for large-scale evaluation of scientific WSD systems. To overcome this obstacle, we revisit pseudowords as a technique to create an artificial dataset to match our needs.

4.2 Pseudowords Revitalized

In this section, we will describe the steps we used to constructed our artificial WSD dataset.

4.2.1 Term Selection

A critical component of creating a pseudowords dataset is having a set of unambiguous terms, which can be used as pseudosenses. Additionally, we are interested in analyzing performance on scientific terms in particular, so we needed to collect a large amount of unambiguous scientific terms.

To automate this collection process, we decided to scrape pages from Wikipedia since it has a rich set of terms that cover a large number of fields. In order to specifically isolate terms in certain research fields, we only selected terms from particular Wikipedia glossary pages. A list of the glossaries we used can be found in the appendix (A.1).

We processed each term in the glossaries (skipping over terms that contained numbers, punctuation, or non-ASCII characters). We then searched for the terms over a corpus of over 2 million papers in the S2ORC dataset and kept only terms that appeared in at least 100 different abstracts. Pilehvar and Navigli (2014) used a minimum frequency of 1000 occurrences in what is one of the most recent works on building a large-scale pseudowords dataset. However, they were using more common English terms and searching across the much larger English Gigaword corpus (Graff et al., 2003). Thus, we felt it was appropriate to relax the minimum frequency constraint from 1000 to 100. We collected a total of 2517 unambiguous terms.

For the definition of each term, we simply take the entirety of the associated summary in the Wikipedia page for the term. The summaries averaged a length of 209 words. An example is shown in Figure 3.

Term: Flow Velocity
Summary: In continuum mechanics the flow velocity in fluid dynamics, also macroscopic velocity in statistical mechanics, or drift velocity in electromagnetism, is a vector field used to mathematically describe the motion of a continuum. The length of the flow velocity vector is the flow speed and is a scalar. It is also called velocity field; when evaluated along a line, it is called a velocity profile (as in, e.g., law of the wall).

Figure 3: Example of an unambiguous term from Wikipedia and its associated summary.

4.2.2 Dataset Creation

In building our dataset, we need to decide on a few main properties: size, distribution of number of senses, the distribution of senses, and train/test split.

Dataset Size: Our goal is to create a large-scale dataset for robust training. Previous work (Pilehvar and Navigli, 2014) has used 1000 examples per

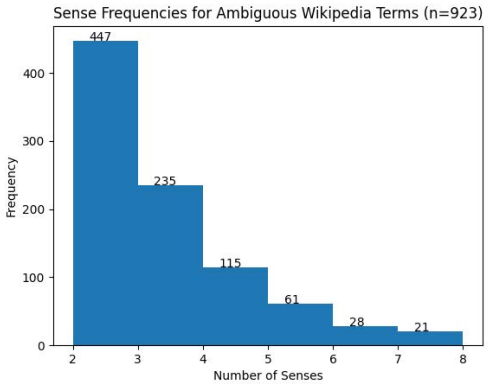


Figure 4: Distribution of number of senses per ambiguous term collected from Wikipedia. The last bin contains terms with seven senses or more.

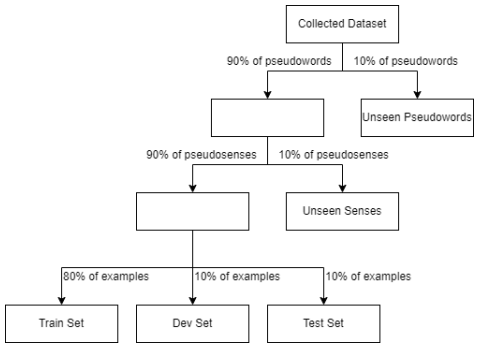


Figure 5: Illustration of the Train-Dev-Test split for the Pseudowords dataset.

pseudoword. However, since we relaxed our minimum frequency, we will also reduce this constraint to only 500.

Distribution of Number of Senses: Next, we want our dataset to have terms with a variety of different number of senses to ensure a trained model is capable of disambiguating terms with various levels of difficulty. To equitably evaluate a model's ability across a different number of senses, we uniformly distribute the number of pseudowords with a certain number of senses ranging from 2 to 8. We limit the number of senses to 8 since prior work (Pilehvar and Navigli, 2014) has shown that the frequency of terms with senses beyond 8 is quite small. We confirm this finding by plotting the distribution of senses for a set of ambiguous Wikipedia terms in Figure 4.

Distribution of Senses: The various senses of a term rarely occur with the same frequency. It is typically the case that one sense will dominate the others in terms of frequency, which is why the most frequent sense is a powerful baseline (Kilgarriff, 2004). We want our dataset to capture this uneven distribution and have models learn to be robust to these biases. Pilehvar and Navigli (2014) collected empirical data on the distribution of senses in SemCor, which we will force our pseudowords to follow.

Train-Test Split In splitting our dataset, we want to not simply evaluate a model's ability to generalize to new examples for senses seen in the training set, but also new senses for a term and new terms as well. Thus, we adopt the following split: we remove 10% of pseudowords from the collected dataset and create a test set for unseen terms, evenly selecting terms of various number of senses. We then remove 10% of all senses and add them to a test set for unseen senses. For the remaining terms, we remove 10% of examples per term and create a test set for these seen terms and senses. We then remove another 10% to create a dev set. This gives us the ability to thoroughly evaluate a model in many scenarios. A diagram of the split is shown in Figure 5.

4.2.3 Pseudoword Formation

Traditionally, non-random pseudoword formation has relied on structured knowledge in the form of task independent resources like WordNet (Pilehvar and Navigli, 2014). However, we do not have access to such a resource in this very general open domain of scientific terms. The reason such resources are useful is that they make it easy to identify similar senses using graph-based techniques for similarity comparison like Personalized PageRank (Haveliwala, 2002). Now, we have access to pre-trained transformers that are capable of producing embedding vectors that are designed to do this similarity comparison without directly relying on structured information. Thus, we propose a pseudoword formulation strategy using BERT-based embedding vectors. So, we first take each of the term definitions, described above, and embed it using a Sentence-BERT model (we use the same model as in 3.2). We then normalize the embeddings to be unit length.

Next, we want to group together vectors in a way that represents the distribution of senses in real ambiguous words. Traditional clustering techniques like K-Means or Agglomerative clustering would produce lots of clusters with only a single term, so we need an approach that will allow us to set the number of terms in a cluster. This will also enable

us to fix the distribution of senses.

We devise the following greedy k-Nearest Neighbor approach. We iterate through our list of unambiguous terms. For each term, we fix a number of terms k to group together. We select these k terms by simply picking the k nearest embedding vectors. However, we do not want to reuse a term as a pseudosense for different pseudowords. Thus, in the case where one of the top k nearest neighbors has already been selected, we look for the next nearest neighbor. This approach is more clearly defined in Algorithm 1.

A drawback of this approach is that many of the terms in a cluster could be so closely related it would be quite difficult to distinguish them. To form pseudowords that more closely model real ambiguous words, we investigate the sense distribution of ambiguous terms. In order to generate a model that would fit the terms in our pseudowords dataset, we specifically select ambiguous terms from the same Wikipedia categories we selected our unambiguous terms from. To do this, we scraped the same Wikipedia glossary pages described earlier, except now we filtered for ambiguous terms. We collected the senses of a term by following the links on a term's disambiguation page. We filtered out a variety of senses using a heuristic of simple string matching since a lot of terms have references to pop culture/arts that we did not want to consider. A list of the strings we filtered out are in the appendix (A.2). We also wanted to filter out related terms that were not actually homonyms, which we could do via a simple regex (also shown in the appendix).

Now, given this set of ambiguous terms, we take all of the senses for a term and embed them using the Sentence-BERT model. We then compute pairwise euclidean distances for all of the senses. We do this for all of the collected ambiguous terms. We take the median of the minimum distances and adjust the greedy approach to favor points that are closer to this threshold. We chose the median of the minimum values since the goal of this step is to ensure that the disambiguation is not too hard or impossible, so our priority is to ensure the distribution of the minimum distances are sufficiently large.

We then plot the distribution of the minimum distances of the real ambiguous words, our pseudowords before the adjustment, and after using a box plot as shown in Figure 6. A key takeaway is

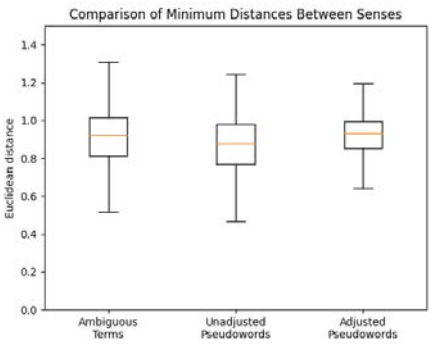


Figure 6: This chart compares the distribution of the minimum distance between two senses for real ambiguous words, our unadjusted pseudowords, and our adjusted pseudowords.

that our adjusted median minimum is closer to the median minimum of the real ambiguous terms and the distribution is tighter making all words more comparable. We also investigated the distribution of other relevant order statistics such the the maximums, medians, and means. However, we didn't use these values in our adjustment here. We will leave investigating this to future work, but will include plots of these distributions in the appendix (A.3).

It is an open question on how we can cluster terms such that we better fit the distribution of real ambiguous terms. Prior work (Pilehvar and Navigli, 2014) has attempted this by picking pseudo-senses that are similar to the senses of the ambiguous word, thus, directly fitting to the distribution of real ambiguous words. One major difficulty in accomplishing this in our use case is having enough examples per sense since the number of words with a large number of senses is low. Thus, we will leave experimenting with this approach using BERT vectors to future work.

We now have a complete process for generating pseudowords. An example of a generated pseudowords and its senses is shown in Figure 7.

Pseudoword: MSP-246
Senses: Monte Carlo Method, Stochastic Process, Probability Theory

Figure 7: Example of a generated pseudoword and its senses. The actual pseudoword is generated by concatenating the first character of each sense and then appending a unique identifier number.

Algorithm 1 Greedy approach for pseudoword formation

```
procedure PSEUDOWORDFORMATION
    numberOfSenses ← [2,3,4,5,6,7,8,2...]          ▷ Uniform distribution of number of senses
    pseudoDictionary ← Map()
    selectedSenses ← Set()
    term ← next(psesudoSenses)
    for k ∈ numberOfSenses do
        while term in selectedSenses do
            term ← next(psesudoSenses)
        group ← []
        neighbors ← getNN(term)      ▷ Returns terms sorted by distance. Nearest neighbor is itself
        for neighbor ∈ neighbors do
            if neighbor not in selectedSenses then          ▷ Only consider unselected terms
                group.append(neighbor)
                selectedSenses.add(neighbor)
            else if size(group) == k then
                break
        pseudoword ← getPseudoword(group)      ▷ Create pseudoword based on grouped terms
        pseudoDictionary[pseudoWord] ← group
    return pseudoDictionary
```

4.3 Results

We now evaluate our STARDUST model on our pseudowords dataset. We also compare it against a highly performing English WSD model, BEM (Blevins and Zettlemoyer, 2020).

We train the BEM model on our train dataset for 10 epochs using a learning rate of 1e-5, the Adam optimizer (Kingma and Ba, 2015), a warmup of 10000 steps, a linear decay, and a batch size of 2. For the sake of efficiency, we tie the weights of the gloss and context encoder since the paper mentioned that there is little performance differences in doing so. We initialize the encoders using BERT-base. It takes about 4.5 days to train on a NVIDIA Titan RTX.

We train our base STARDUST model, with no context enhancement or citation graph information, on the train set as well. However, we keep the encoder models frozen and only training the classifier head using a learning rate of 1e-3, while all the other hyperparameters were the same as BEM. It takes about 2 days to train on a NVIDIA Titan RTX.

We then test our model on the three test sets described above: In-Train, New Senses, New Words. The results can be in Table 5.

As we can see from the table, STARDUST doesn't perform as well as BEM on the test set that has the same distribution as the training set.

Model	Test Set		
	In-Train	New Senses	New Words
BEM	**0.7634**	0.0799	0.4419
BEM*	0.7625	0.3445	0.4386
STARDUST	0.5568	**0.5374**	**0.4932**

Table 5: F1 scores of the BEM and STARDUST models on our three Pseudoword WSD test sets.

This is expected since the BEM model is finetuning its encoders on the training dataset, using many more parameters. However, STARDUST significantly outperforms BEM on the other two test sets that analyze the generalizability of the models.

The New Senses test set contains examples using the same words as in the training set, however the correct senses were never used as labels in the training set. BEM performs quite poorly in this setting. A possible reason for this is that it still sees the same definitions while training, but since those definitions are never assigned to a training example, the encoders probably learned to bias against them when scoring. STARDUST, on the other hand, doesn't finetune its encoders, so this same bias can't be as easily learned, which explains this significant boost. The justification behind this type of split is that there do exist definitions of terms that we know about, but there might not be

any labeled examples. We would ideally want our models to learn to generalize to these unlabeled examples without having to change the dictionary. We also see how BEM compares when it doesn't see these definitions while training (BEM*). As expected, performance drastically increases (while similar on the other test sets), but still falls short of STARDUST.

On the New Words test set, STARDUST performs about 5 F1 points above BEM, once again speaking to its ability to strongly generalize to new words and definitions. This gap is much smaller than the New Senses test set, but still significant given its learning significantly fewer parameters. Part of this can also be attributed to using SBERT as the encoder, which has shown to produce stronger representations, but BEM finetunes its BERT encoders, which should make them comparable.

Overall, STARDUST represents the strong performance benefits that come with giving specific attention to the type of documents at hand when performing WSD. Generaziliability towards infrequent senses is the core challenge in WSD and this architecture is showing massive improvements in this direction.

However, much further work is needed to boost these scores as they are still quite low overall. One potential next step is to finetune the encoders in the STARDUST model and see if generalizability is still able to be maintained while also improving performance on the in train distribution test set, at the cost of using many more parameters. Additionally, we can also explore using context enhancement and more techniques for using the citation graph.

5 Conclusion

In this paper, we investigated the use of auxiliary paper information for tasks related to Scientific Sense Disambiguation, specifically Acronym Disambiguation and Word Sense Disambiguation. We propose STARDUST, a novel tri-encoder system that can combine information about a sense, a word, and a scientific paper into a disambiguation decision. This technique demonstrates significant improvement on AD over methods that forgo this additional information.

We also present a revitalization of the pseudowords technique to create a modern large-scale dataset for WSD in Scientific Text, enabling researchers to further the work in this space.

Our method performs competitively against high-performing methods designed for general English WSD, specifically in terms of generalizability, demonstrating the benefits of providing specific attention to scientific papers in designing WSD systems.

However, performance is still quite lacking for infrequent and unseen words/senses, where methods are performing 30-50 points worse compared to words/senses seen during training. Closing this performance gap is critical within scientific fields due to their fast growing vocabularies.

The immediate next steps are exploring finetuning the STARDUST model end-to-end, increasing the number of learned parameters to help it fit better to the training data, as well as using our context enhancement technique to enable more robust representation when doing inference.

Further work can explore employing our techniques for incorporating paper information with methods that balance the training signal for more frequent senses to prevent too much bias towards more frequent senses.

Additionally, more work is needed to understand how we can use the citation graph to better update the representation of a paper. Our approach naively includes are papers within the one-hop citation neighborhood. However, some citations/references are more important than others. Approaches that can figure out a way to leverage this notion might be able to reduce the noise that comes with our naive approach and create a stronger representation.

Our resurrection of pseudowords using modern techniques for similarity comparison also enables a slew of new questions and potential areas of research. First, there is a need for a human evaluation of the quality of our generated pseudowords to validate our technique. Second, we took a simple greedy approach to creating clusters that might not be optimal. So, more work on intelligent clustering strategies is necessary. Next, we attempted to model of pseudoword clusters to real ambiguous words using a very simple heuristic of comparing pairwise distances of embedded senses. More work is needed to understand how we can better align these models. Lastly, we can investigate the viability of this technique in other low-resource domains.

To facilitate research, we release our code and trained models at https://github.com/ManavR123/definition_disambiguation.

References

Edoardo Barba, Tommaso Pasini, and Roberto Navigli. 2021. ESC: Redesigning WSD with extractive sense comprehension. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4661–4672, Online. Association for Computational Linguistics.

Charles Bazerman. 1985. Physicists reading physics: Schema-laden purposes and purpose-laden schema. *Written communication*, 2(1):3–23.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Michele Bevilacqua, Marco Maru, and Roberto Navigli. 2020. Generationary or "how we went beyond word sense inventories and learned to gloss". In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7207–7221, Online. Association for Computational Linguistics.

Terra Blevins and Luke Zettlemoyer. 2020. Moving down the long tail of word sense disambiguation with gloss informed bi-encoders. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1006–1017, Online. Association for Computational Linguistics.

Cornelia Caragea, Florin Adrian Bulgarov, Andreea Godea, and Sujatha Das Gollapalli. 2014. Citation-enhanced keyphrase extraction from research papers: A supervised approach. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1435–1446, Doha, Qatar. Association for Computational Linguistics.

Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel Weld. 2020. SPECTER: Document-level representation learning using citation-informed transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2270–2282, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Andres Duque, Mark Stevenson, Juan Martinez-Romo, and Lourdes Araujo. 2018. Co-occurrence graphs for word sense disambiguation in the biomedical domain. *Artificial intelligence in medicine*, 87:9–19.

Philip Edmonds and Scott Cotton. 2001. SENSEVAL-2: Overview. In *Proceedings of SENSEVAL-2 Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 1–5, Toulouse, France. Association for Computational Linguistics.

Christiane D. Fellbaum. 2000. Wordnet : an electronic lexical database. *Language*, 76:706.

William A. Gale, Kenneth W. Church, and David Yarowsky. 1992a. One sense per discourse. In *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992*.

William A. Gale, Kenneth Ward Church, and David Yarowsky. 1992b. Work on statistical methods for word sense disambiguation.

Krishna Garg, Jishnu Ray Chowdhury, and Cornelia Caragea. 2021. Keyphrase Generation Beyond the Boundaries of Title and Abstract. *arXiv e-prints*, page arXiv:2112.06776.

Tanja Gaustad. 2001. Statistical corpus-based word sense disambiguation: Pseudowords vs. real ambiguous words. In *ACL*.

David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2003. English gigaword. *Linguistic Data Consortium, Philadelphia*, 4(1):34.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Taher H. Haveliwala. 2002. Topic-sensitive pagerank. In *WWW '02*.

Luyao Huang, Chi Sun, Xipeng Qiu, and Xuanjing Huang. 2019. GlossBERT: BERT for word sense disambiguation with gloss knowledge. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3509–3514, Hong Kong, China. Association for Computational Linguistics.

Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondrej Chum. 2019. Label propagation for deep semi-supervised learning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 5070–5079. Computer Vision Foundation / IEEE.

Adam Kilgarriff. 2004. How dominant is the commonest sense of a word? In *TSD*.

Minho Kim and Hyuk chul Kwon. 2021. Word sense disambiguation using prior probability estimation based on the korean wordnet. *Electronics*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Carolyn E. Lipscomb. 2000. Medical subject headings (mesh). *Bulletin of the Medical Library Association*, 88 3:265–6.

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel Weld. 2020. S2ORC: The semantic scholar open research corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4969–4983, Online. Association for Computational Linguistics.

Daniel Loureiro, Alípio Mário Jorge, and Jose Camacho-Collados. 2022. Lmms reloaded: Transformer-based sense embeddings for disambiguation and beyond. *Artificial Intelligence*, page 103661.

Zhimao Lu, Haifeng Wang, Jianmin Yao, Ting Liu, and Sheng Li. 2006. An equivalent pseudoword solution to Chinese word sense disambiguation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 457–464, Sydney, Australia. Association for Computational Linguistics.

George A. Miller, Martin Chodorow, Shari Landes, Claudia Leacock, and Robert G. Thomas. 1994. Using a semantic concordance for sense identification. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

Takeru Miyato, Andrew M. Dai, and Ian J. Goodfellow. 2017. Adversarial training methods for semi-supervised text classification. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Andrea Moro and Roberto Navigli. 2015. SemEval-2015 task 13: Multilingual all-words sense disambiguation and entity linking. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 288–297, Denver, Colorado. Association for Computational Linguistics.

Preslav Nakov, Ariel S. Schwartz, and Marti A. Hearst. 2004. Citances: Citation sentences for semantic analysis of bioscience text.

Preslav I. Nakov and Marti A. Hearst. 2003. Category-based pseudowords. In *Companion Volume of the Proceedings of HLT-NAACL 2003 - Short Papers*, pages 67–69.

Roberto Navigli, David Jurgens, and Daniele Vannella. 2013. SemEval-2013 task 12: Multilingual word sense disambiguation. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 222–231, Atlanta, Georgia, USA. Association for Computational Linguistics.

Lubomir Otrusina and Pavel Smrz. 2010. A new approach to pseudoword generation. In *LREC*.

Chunguang Pan, Bingyan Song, Shengguang Wang, and Zhipeng Luo. 2021. Bert-based acronym disambiguation with multiple training strategies. *arXiv preprint arXiv:2103.00488*.

Mohammad Taher Pilehvar and Roberto Navigli. 2014. A large-scale pseudoword-based evaluation framework for state-of-the-art word sense disambiguation. *Computational Linguistics*, 40(4):837–881.

Amir Pouran Ben Veyseh, Franck Dernoncourt, Walter Chang, and Thien Huu Nguyen. 2021. MadDog: A web-based system for acronym identification and disambiguation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 160–167, Online. Association for Computational Linguistics.

Amir Pouran Ben Veyseh, Franck Dernoncourt, Quan Hung Tran, and Thien Huu Nguyen. 2020. What does this acronym mean? introducing a new dataset for acronym identification and disambiguation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3285–3301, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Sameer Pradhan, Edward Loper, Dmitriy Dligach, and Martha Palmer. 2007. SemEval-2007 task-17: English lexical sample, SRL and all words. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 87–92, Prague, Czech Republic. Association for Computational Linguistics.

Roman Prokofyev, Gianluca Demartini, Alexey Boyarsky, Oleg Ruchayskiy, and Philippe Cudré-Mauroux. 2013. Ontology-based word sense disambiguation for scientific literature. In *European conference on information retrieval*, pages 594–605. Springer.

Alessandro Raganato, Jose Camacho-Collados, and Roberto Navigli. 2017. Word sense disambiguation: A unified evaluation framework and empirical comparison. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 99–110, Valencia, Spain. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*

and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Benjamin Snyder and Martha Palmer. 2004. The English all-words task. In *Proceedings of SENSEVAL-3, the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 41–43, Barcelona, Spain. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15:1929–1958.

Carol Tenopir and Donald W. King. 2008. Electronic journals and changes in scholarly article seeking and reading patterns. *Aslib Proc.*, 61:5–32.

Nemika Tyagi, Sudeshna Chakraborty, Aditya Kumar, Nzanzu Katasohire Romeo, et al. 2022. Word sense disambiguation models emerging trends: A comparative analysis. In *Journal of Physics: Conference Series*, volume 2161, page 012035. IOP Publishing.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Amir Pouran Ben Veyseh, Franck Dernoncourt, Thien Huu Nguyen, Walter Chang, and Leo Anthony Celi. 2021. Acronym identification and disambiguation shared tasks for scientific document understanding. *ArXiv*, abs/2012.11760.

Vijay Viswanathan, Graham Neubig, and Pengfei Liu. 2021. CitationIE: Leveraging the citation graph for scientific information extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 719–731, Online. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

aerospace engineering	agriculture
architecture	artificial intelligence
biology	bird terms
botanical terms	chemistry terms
climate change	computer_science
ecology	economics
genetics	geography terms
geology terms	graph theory
history terms	mechanical engineering
medicine	meteorology
philosophy	physics
probability and statistics	virology

Figure 8: List of glossaries we used to select terms for our pseudowords dataset.

A Pseudowords

A.1 Term Selection

As mentioned in section 4.2.1, we hand-picked several Wikipedia glossaries, to scrape terms from for our pseudowords dataset. A list of these glossaries can be found in Figure 8.

A.2 Sense Filtering

When seeking out ambiguous terms to understand the true distribution of senses distances, we filtered our a variety of Wikipedia senses that we believed to be noisy i.e. senses that were just movies, songs, etc. We filtered out these senses by a simple string matching approach, which works effectively given how Wikipedia tends to label terms by adding a descriptor inside a set of parentheses e.g. Transformer (film). A list of all of the strings we filtered out can be found in Figure 9. We also filtered out senses that contained non-ascii characters, numbers, and linked pages that did not contain the original term since Wikipedia disambiguation pages often link to pages that are simply related, but not actually homonyms. We enforced the final constraint by using a regex that matches only to homonyms. An example of the regex can be seen in Figure 10.

A.3 Sense Distribution

In section 4.2.3, we describe a process for adjusting the distribution of pseudoword senses in the embedding space by matching them to the distribution of distances of real ambiguous words. However, in that section we only had a discussion of the minimum values and left out looking at other potentially relevant information. In this section, we include figures that also show information about

(film)	(band)	(album)
(musical)	(song)	(music)
(game)	(novel)	(comic)
(name)	(disambiguation)	(story)
(series)	(entertainment)	(play)
(character)	(composition)	(magazine)
(ep)	(movie)	(serial)
(journal)	(channel)	the
(actor)	(records)	(software)
river	lake	
:	"	,

Figure 9: We filtered out senses of ambiguous terms that contained any of the above words/punctuation. These were manually selected by manually inspecting the collected data and identifying unwanted senses.

Term: height
Senses: height (abelian group), tree height, height of a field
Regex: height \([a-z\s]+\)
Matches: height (abelian group)

Figure 10: An example of the regex we used to filter for homonyms.

the maximums, medians, means, and overall distribution of sense distances. Figure 11 shows the broader distribution of distances of real ambiguous words. Figures 12 and 13 show the before and after distance distributions of our pseudowords.

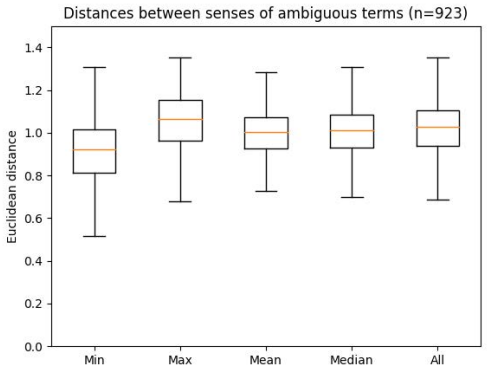


Figure 11: Distribution of distances of real ambiguous words scraped from Wikipedia. The distribution of the number of senses per term can be found in Figure 4.

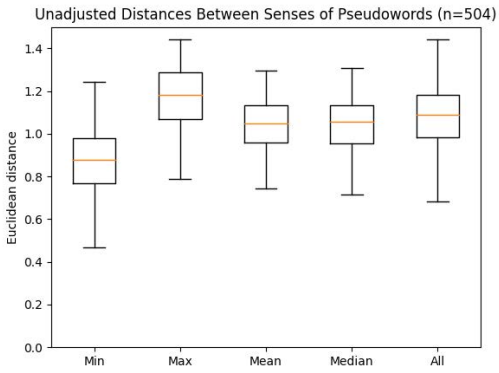


Figure 12: Distribution of distances of pseudowords before applying adjustment.

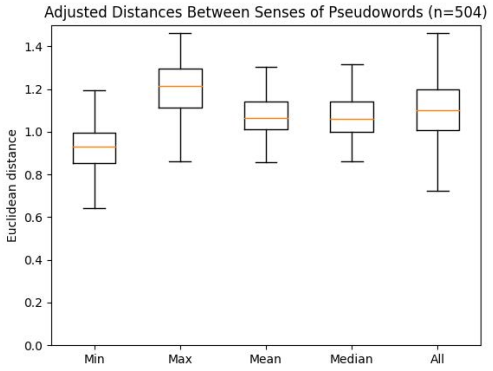


Figure 13: Distribution of distances of pseudowords after applying adjustment.