BERT Has Uncommon Sense: Similarity Ranking for Word Sense BERTology

Luke Gessler **Nathan Schneider**
Georgetown University
{lg876, nathan.schneider}@georgetown.edu

Abstract

An important question concerning contextualized word embedding (CWE) models like BERT is how well they can represent different word senses, especially those in the long tail of uncommon senses. Rather than build a WSD system as in previous work, we investigate contextualized embedding neighborhoods directly, formulating a query-by-example nearest neighbor retrieval task and examining ranking performance for words and senses in different frequency bands. In an evaluation on two English sense-annotated corpora, we find that several popular CWE models all outperform a random baseline even for proportionally rare senses, without explicit sense supervision. However, performance varies considerably even among models with similar architectures and pretraining regimes, with especially large differences for rare word senses, revealing that CWE models are not all created equal when it comes to approximating word senses in their native representations.

1 Introduction

Contextualized word embedding (CWE) models such as BERT (Devlin et al., 2019), which were enabled by Transformers (Vaswani et al., 2017), have yielded great improvements in a variety of NLP tasks. However, because BERT and other CWE models are deep neural networks with complicated architectures and very high parameter counts, it is not easy to understand exactly which aspects of linguistic form and meaning contextualized word embeddings are able to capture.

In response to these difficulties, much work has been done attempting to interpret CWE models, most notably in the field of BERTology. Over 100 BERTological studies have been published since BERT was introduced in 2018, covering a diverse range of linguistic phenomena (Rogers et al., 2020) such as POS tags, constituency, and event factuality (Liu et al., 2019a). The methods developed in this area are usually applicable to other CWE models.

An important question for CWE models is how well they can represent rare word senses. Word sense disambiguation systems have been observed to be most lacking in the long tail of rare word senses (Blevins and Zettlemoyer, 2020; Blevins et al., 2021), and sense-awareness is of fundamental importance for many NLP applications. Thus a better understanding of how well CWE models understand senses, and especially rare senses, would aid the interpretation of NLP systems.

To address this question, we perform an evaluation we call CWE similarity ranking on two sense-annotated English corpora using several popular CWE models. We find that while all models outperform a random baseline on the evaluation, models differ substantially in their performance on rare word senses, with significant differentiation even between models which are closely related.[1]

2 Previous Work

2.1 Word Senses

The task of word sense disambiguation (WSD) introduced the notion of a *word sense* into NLP (Navigli, 2009; Bevilacqua et al., 2021). The WSD task is typically formulated as labeling words in context with their senses as defined by a dictionary or other lexical resource.

Many resources exist to support work on word senses. WordNet (Miller, 1992) has been a crucial resource for work on word senses, providing a fine-grained and comprehensive inventory of words and their senses for English. Several large annotated corpora have been constructed using WordNet senses, including SemCor (Miller et al., 1993; Landes et al., 1998) and OntoNotes (Hovy et al., 2006). More recently, WSD systems

[1] All code for this work is available at https://github.com/lgessler/bert-has-uncommon-sense.

Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP, pages 539–547

have been able to achieve human-like performance on WSD tasks, but performance on rare senses remains comparatively poor (Blevins and Zettlemoyer, 2020), leading to the construction of corpora tailored for assessing systems on rare senses (Blevins et al., 2021).

2.2 Word Sense BERTology

Few studies have focused narrowly on the question of how well BERT and other CWE models capture word senses. In one such study, Wiedemann et al. (2019) evaluate CWE models by using the CWE models' embeddings as representations for a kNN classifier: the predicted sense of a word is the one most represented among by the word's k nearest neighbors (via cosine distance). Despite the simplicity of this WSD system, it is able to achieve results that are competitive with state-of-the-art systems, even achieving new SOTA scores on the SensEval-2 and SensEval-3 datasets. Reif et al. (2019) construct a similar WSD system, relying not on kNN but closest sense-centroids (centroids are constructed using a labeled training set) to decide on a label.

Other studies have approached the question by modifying BERT's training scheme. Tayyar Madabushi et al. (2020) train a BERT variant where the next sentence prediction task has been replaced with a same construction prediction task and find mixed results on downstream tasks. Levine et al. (2020) train a BERT variant by adding a new supersense prediction task, wherein the masked token's WordNet supersense is to be predicted, and find performance gains on a variety of meaning-related tasks, which shows that BERT's representations do not perfectly capture word senses.

Some recent approaches have developed specialized methods for exploring CWE models' embedding spaces. Karidi et al. (2021) present an iterative method for surveying the "topography" of BERT's embedding space, finding that word senses often form cohesive regions.

3 CWE Similarity Ranking

We formulate a query-by-example task in which a word in sentence context is used to query for similar usages of the same word in other sentences. In a process we call **contextualized word embedding (CWE) similarity ranking**, we will assume some embedding function f and two corpora of sense-labeled text segmented into sentences: a larger "database" corpus $\mathcal{D}$, and a smaller "query" corpus $\mathcal{Q}$. We will use sentences from $\mathcal{Q}$ to rank sentences in $\mathcal{D}$, as detailed below.

1. Select a query sentence from $\mathcal{Q}$ consisting of tokens $w_1^{(q)}, \ldots, w_n^{(q)}$, where a token $w_i^{(q)}$ has been designated as the *target token* and has sense s_i.
2. Find embeddings for the query sentence, $\mathbf{h}_1^{(q)}, \ldots, \mathbf{h}_n^{(q)} = f\left(w_1^{(q)}, \ldots, w_n^{(q)}\right)$.
3. For every instance $w_j^{(d)}$ with its context $w_1^{(d)}, \ldots, w_m^{(d)}$ in $\mathcal{D}$, and sense s_j, find embeddings $\mathbf{h}_1^{(d)}, \ldots, \mathbf{h}_m^{(d)} = f\left(w_1^{(d)}, \ldots, w_m^{(d)}\right)$.
4. Rank instances in $\mathcal{D}$ in descending order by cosine similarity between embeddings of the two tokens, $w_i^{(q)}$ and $w_j^{(d)}$: $\textsc{cos-sim}\left(\mathbf{h}_i^{(q)}, \mathbf{h}_j^{(d)}\right)$.
5. Evaluate the ranking, e.g. with precision at k.

CWE similarity ranking consists of much of the same work that a CWE-based kNN system would do,[2] with the difference that the kNN WSD evaluation will only reward a model if the gold sense is held by a plurality of the neighbors, whereas CWE similarity ranking is less stringent and will award "partial credit" for retrieving any gold instances, even if they do not form a majority. This provides a clearer view of how coherent rare word senses are in CWEs' representations. See Figure 1 for query examples.

4 Experimental Setup

We use the CWE similarity ranking method to evaluate several popular pretrained CWE models retrieved from `huggingface.co` (Wolf et al., 2020).

Corpora Our two corpora are OntoNotes 5.0 (Hovy et al., 2006), which has sense annotations for nouns and verbs,[3] and the Pattern Dictionary of English Prepositions (PDEP) corpus (Litkowski, 2014), which has sense annotations for prepositions.[4] For OntoNotes, we discard instances labeled with "none-of-the-above" senses, as we expect them to be heterogeneous. For PDEP, we

[2] Indeed, the basic method of finding embeddings that are nearest to a target embedding has been widely used both before and after the rise of CWEs (cf. Wiedemann et al. 2019, which we describe in §2.2, and Schnabel et al. 2015).

[3] Specifically, we use the OntoNotes English noun and verb *sense groupings*, whose sense inventory was formulated by merging WordNet senses for each lemma until an acceptable level of interannotator agreement was reached. Our preliminary experiments with WordNet senses in another corpus, SemCor (Langone et al., 2004), were difficult to interpret because annotations often seemed to be inconsistent.

[4] We use a copy of PDEP obtained from a SQL dump dated 2019-04-19. Original data is available with our code.

Query: But with all the money and glamour of high finance come the relentless pressures to do well; pressure to **pull**$_4$ off another million before lunch [...]
1: "Sometimes," he says, "we'll **pull**$_3$ someone off phones for more training."
2: Hence, they have never lacked their own stately or amusing charms to **pull**$_2$ in wealth and keep it within a household.
3: I can't **pull**$_4$ it off.
4: Bulatovic says Kostunica was able to **pull**$_4$ off the balancing act because he is not really anti-American.

(a) A sample of a single query on OntoNotes for the verb *pull*. Sense 4, glossed as 'commit, do', is rare, comprising just 5 of the 78 occurrences of *pull* in the OntoNotes training set. Here, the first two results are incorrect: sense 3 means 'eliminate from a situation', and sense 2 means 'steer something in a certain direction'. All 5 correct instances are retrieved in the top 50 results, at ranks 3, 4, 5, 6, and 41.

Query: These days I take five pills a day, but at one point I was **on**$_{20}$ about 20.
1: I'm happier than I was three years ago, when I was drinking and I was **on**$_{20}$ cocaine.
2: I was **on**$_{20}$ about sixty cigarettes a day.
3: I was **on**$_{20}$ heavy duty painkillers for 48 hours.
4: He had been put **on**$_{20}$ prescription drugs to help him cope with coming off crack.
5: Recorded in 21 days in a Mitcham garage, the fact that it is **on**$_2$ Chrysalis is a mere coincidence.
6: You can phone now **on**$_{12}$ o-five-hundred , four-o-four , treble zero to put your views to Tory MP Phil Gallie .

(b) A sample of a single query on PDEP for the preposition *on*. Sense 20, glossed as 'regularly taking (a drug or medicine)', is very rare, comprising 14 of the 1728 occurrences of *on* in the PDEP training set. The 5th and 6th results are incorrect: sense 2 has to do with location, and sense 12 a medium of communication. Only 6 of the 14 total occurrences in $\mathcal{D}$ make it into the top 50 for this query.

Figure 1: Query samples using `bert-base-cased`.

use only instances of 48 common English prepositions. For both corpora, we use only single-word targets, and use pre-existing train–validation–test splits. We treat the training split of each as our $\mathcal{D}$, and the combined validation and test split of each as our $\mathcal{Q}$, and discard any senses in $\mathcal{Q}$ that did not occur at least 5 times in $\mathcal{D}$, as these senses are so rare in $\mathcal{D}$ that performance on them is liable to be overtaken by noise. For OntoNotes, $|\mathcal{D}|$ = 229,989, $|\mathcal{Q}|$ = 50,395. For PDEP, $|\mathcal{D}|$ = 33,090, $|\mathcal{Q}|$ = 8,020.

Inoculation by Fine-Tuning In addition to the base models, we also evaluate versions of the models that have been fine-tuned with a small number of instances from another dataset, a method called *inoculation by fine-tuning* (Richardson et al., 2020; Liu et al., 2019b). Inoculation by fine-tuning allows model to surface more domain-relevant information in its output embeddings, while using only a small amount of data so as to avoid teaching the model anything entirely new. We use STREUSLE 4.4 (Schneider and Smith, 2015; Schneider et al., 2018) to fine-tune, sampling supersense annotations of single-word nouns, verbs and prepositions in equal numbers for total counts of 100, 250, 500, 1000, and 2500. See Appendix A for full details.

Layer Choice In this work, we use embeddings from the last layer of every model we assess. Preliminary experiments showed that many models show no improvement past the middle layers, and using the last layer is also of interest because many systems freeze their CWE models' weights and use embeddings from their final layers.

Lemma Restriction In all cases, neighbors are restricted to instances that have the same lemma.

Evaluation To assess the quality of a ranking, we use mean average precision[5] for the top 50 ranked instances.[6] Since there can be very few gold-labeled instances, it may be impossible for a model to achieve a perfect score. To make these metrics more interpretable for a given dataset, we also include a baseline score obtained by randomly ranking results, and an oracle upper bound, i.e. the score that would be obtained by a perfect model.[7]

5 Results

Main results are given in Table 1. Instances are bucketed by two parameters: ℓ is the instance's lemma's frequency in $\mathcal{D}$, and r is the proportion of all instances of the lemma that have the same sense in $\mathcal{D}$. Recall that each cell in Table 1 is *mean* average precision over precision at k for 50 instances, and see Figure 2 for a sample of what the P@K curves look like for each cell in Table 1.

Two example queries and top results appear in Figure 1. Consider the results for the "on" query at right: out of a large haystack (1728 available tokens of "on"), the system has correctly retrieved 6 of the 14 relevant needles, featuring 4 at the top of the list! Interestingly, even though the complement of *on* in the query instance is just "20" (a fused-head construction; Elazar and Goldberg, 2019) instead

[5] Recall is omitted here—see Appendix B.

[6] Average precision is defined for a single query as the average of precision calculated for every result from 1 to some k, where k ranges from 1 to a maximum of 50 for the present study. Mean average precision in turn is the average of this quantity across all instances in $\mathcal{Q}$.

[7] Note that this is not always 100%: for queries with fewer than 50 gold instances that can be retrieved in all of $\mathcal{D}$, the oracle's performance will be under 100%.

Model	$\ell < 500$ $r < 0.25$	$\ell < 500$ $r \geq 0.25$	$\ell > 500$ $r < 0.25$	$\ell > 500$ $r \geq 0.25$
Baseline	11.55	62.41	9.55	76.08
Oracle	82.02	93.89	100.00	100.00
bert-base-cased	41.60	81.89	48.48	88.53
distilbert-base-cased	39.80	81.32	48.17	88.50
roberta-base	32.87	78.39	45.16	88.37
distilroberta-base	29.33	76.48	43.69	86.86
albert-base-v2	40.44	81.81	51.58	89.56
xlnet-base-cased	28.72	75.07	36.16	84.29
gpt2	18.34	69.56	33.53	82.74

(a) Performance for OntoNotes, no fine-tuning. Buckets respectively contain 6,949, 30,694, 1,649, and 11,123 query instances.

Model	$\ell < 500$ $r < 0.25$	$\ell < 500$ $r \geq 0.25$	$\ell > 500$ $r < 0.25$	$\ell > 500$ $r \geq 0.25$
Baseline	11.56	56.34	18.25	49.88
Oracle	96.41	100.00	100.00	100.00
bert-base-cased	59.59	83.54	66.34	89.39
distilbert-base-cased	58.06	83.15	65.09	88.10
roberta-base	39.84	76.79	47.65	80.01
distilroberta-base	32.42	72.22	42.29	70.81
albert-base-v2	56.53	82.22	66.64	88.44
xlnet-base-cased	35.76	74.08	37.68	75.16
gpt2	21.75	63.41	33.33	61.00

(b) Performance for PDEP, no fine-tuning. Buckets respectively contain 4,970, 1,618, 733, and 699 query instances.

Model	$\ell < 500$ $r < 0.25$	$\ell < 500$ $r \geq 0.25$	$\ell > 500$ $r < 0.25$	$\ell > 500$ $r \geq 0.25$
Baseline	11.55	62.41	9.55	76.08
Oracle	82.02	93.89	100.00	100.00
bert-base-cased	43.42	82.37	49.81	89.45
distilbert-base-cased	41.62	81.98	50.31	89.43
roberta-base	37.87	80.68	53.65	89.43
distilroberta-base	34.74	79.27	48.50	88.74
albert-base-v2	39.26	81.50	51.65	89.31
xlnet-base-cased	37.53	79.12	51.40	87.97
gpt2	18.12	68.92	32.99	82.08

(c) Best performance across all fine-tuning trials for each model on OntoNotes.

Model	$\ell < 500$ $r < 0.25$	$\ell < 500$ $r \geq 0.25$	$\ell > 500$ $r < 0.25$	$\ell > 500$ $r \geq 0.25$
Baseline	11.56	56.34	18.25	49.88
Oracle	96.41	100.00	100.00	100.00
bert-base-cased	60.37	83.49	67.87	89.28
distilbert-base-cased	58.33	82.91	67.27	87.52
roberta-base	48.99	80.32	60.04	85.72
distilroberta-base	42.25	77.25	53.37	79.19
albert-base-v2	53.75	81.85	65.57	86.97
xlnet-base-cased	49.46	80.50	56.19	84.53
gpt2	21.53	63.00	35.57	61.08

(d) Best performance across all fine-tuning trials for each model on PDEP.

Table 1: Mean average precision performance broken down by corpus and model. Performance was further measured on different buckets of instances, as indicated in column headers: ℓ is a query instance's lemma frequency in $\mathcal{D}$, and r is the proportional frequency of a query instance's sense across all instances of the lemma in $\mathcal{D}$.

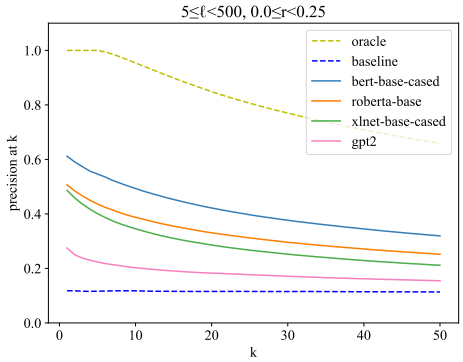

Figure 2: A sample of averaged precision at k curves, showing performance of several models on OntoNotes for the bucket with lemmas occurring fewer than 500 times in $\mathcal{D}$ and senses with proportional frequencies lower than 0.25. Mean average precision, as shown in each cell in Table 1, is obtained by averaging every point along one of these lines.

of the name of a drug or substance, the first four results are still relevant. From examining the top 50 results as well as the 8 false negatives (which include "high on drugs", "drunk on Scotch", etc.), it appears that BERT is prioritizing syntactic criteria ("on" following a verb, especially a copula), possibly because the information following the query preposition is semantically impoverished. In fact, the top 3 results have a trigram match ("I was **on**").

All models perform well above the random baseline across all buckets, and show little differentiation for buckets with only non-proportionally-rare senses $r \geq 0.25$. Dramatic differences emerge, however, for buckets with proportionally rare senses $r < 0.25$, with GPT-2 (Radford et al., 2019) showing the poorest performance. In line with general trends, the distilled variants of models (Sanh et al., 2020) generally track only a few points behind their undistilled variants. Overall, `bert-base-cased` performance is best. ALBERT (Lan et al., 2020) usually performs similarly, and RoBERTa (Liu et al., 2019c) and XLNet (Yang et al., 2019) show noticeably worse performance on rare senses.

It is especially surprising that RoBERTa performs so much worse than BERT on rare senses

(cf. Table 1a, column 1), even though RoBERTa outperforms BERT on GLUE (Wang et al., 2019), and RoBERTa is more closely related to BERT than XLNet or ALBERT: whereas XLNet and ALBERT differ architecturally from BERT, RoBERTa is architecturally identical and differs only in details of training. (For example, it implements dynamic masking, does not perform next sentence prediction, and uses an order of magnitude more data.) Fine-tuning allows RoBERTa to close much of this gap but not all of it (cf. Table 1c). This indicates that word sense information is less readily accessible in RoBERTa embeddings compared to BERT embeddings, and that fine-tuning cannot draw out this information to the extent that it already is in BERT. Taken together, these results call into the question how conclusively a model can be judged based just on performance on downstream task benchmarks: RoBERTa performs better on GLUE, which has led many to prefer it as generally superior to BERT, but we have seen here it seems to be inferior within the domain of lexical semantics.

6 Conclusion

We have presented an evaluation method for probing the word sense content of contextualized word embeddings and applied it to several popular CWE models. We find that their ability to capture rare word senses using their representations is variable and not easily explained by each model's training and architectural characteristics. Moreover, we find that performance of models on this word sense evaluation is not directly proportional to their performance on downstream task benchmarks like GLUE. We view these differences between models as reason for further evaluation of CWE models to investigate their word sense representations, and inquiry into factors that affect word sense content of contextualized word embeddings.

Acknowledgments

We thank Ken Litkowski for providing us with the PDEP corpus and for allowing us to distribute it with our code.

References

Michele Bevilacqua, Tommaso Pasini, Alessandro Raganato, and Roberto Navigli. 2021. Recent trends in word sense disambiguation: A survey. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 4330–4338. International Joint Conferences on Artificial Intelligence Organization. Survey Track.

Terra Blevins, Mandar Joshi, and Luke Zettlemoyer. 2021. FEWS: Large-scale, low-shot word sense disambiguation with the dictionary. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 455–465, 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 *Proc. of ACL*, pages 1006–1017, 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 *Proc. of NAACL-HLT*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Yanai Elazar and Yoav Goldberg. 2019. Where's my head? Definition, data set, and models for numeric fused-head identification and resolution. *Transactions of the Association for Computational Linguistics*, 7:519–535.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. OntoNotes: The 90% solution. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 57–60, New York City, USA. Association for Computational Linguistics.

Taelin Karidi, Yichu Zhou, Nathan Schneider, Omri Abend, and Vivek Srikumar. 2021. Putting words in BERT's mouth: Navigating contextualized vector spaces with pseudowords. In *Proc. of EMNLP*, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *Proc. of ICLR*. OpenReview.net.

Shari Landes, Claudia Leacock, and Randee I. Tengi. 1998. Building semantic concordances. In Christiane Fellbaum, editor, *WordNet: An Electronic Lexical Database*, pages 199–216. MIT Press, Cambridge, MA.

Helen Langone, Benjamin R. Haskell, and George A. Miller. 2004. Annotating WordNet. In *Proceedings of the Workshop Frontiers in Corpus Annotation at HLT-NAACL 2004*, pages 63–69, Boston, Massachusetts, USA. Association for Computational Linguistics.

Yoav Levine, Barak Lenz, Or Dagan, Ori Ram, Dan Padnos, Or Sharir, Shai Shalev-Shwartz, Amnon Shashua, and Yoav Shoham. 2020. SenseBERT: Driving some sense into BERT. In *Proc. of ACL*, pages 4656–4667, Online. Association for Computational Linguistics.

Ken Litkowski. 2014. Pattern dictionary of English prepositions. In *Proc. of ACL*, pages 1274–1283, Baltimore, Maryland. Association for Computational Linguistics.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019a. Linguistic knowledge and transferability of contextual representations. In *Proc. of NAACL-HLT*, pages 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Nelson F. Liu, Roy Schwartz, and Noah A. Smith. 2019b. Inoculation by fine-tuning: A method for analyzing challenge datasets. In *Proc. of NAACL-HLT*, pages 2171–2179, Minneapolis, Minnesota. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019c. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv:1907.11692 [cs]*. ArXiv: 1907.11692.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *Proc. of ICLR*. OpenReview.net.

George A. Miller. 1992. WordNet: A lexical database for English. In *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992*.

George A. Miller, Claudia Leacock, Randee Tengi, and Ross T. Bunker. 1993. A semantic concordance. In *Human Language Technology: Proceedings of a Workshop Held at Plainsboro, New Jersey, March 21-24, 1993*.

Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM Computing Surveys*, 41(2):1–69.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Emily Reif, Ann Yuan, Martin Wattenberg, Fernanda B. Viégas, Andy Coenen, Adam Pearce, and Been Kim. 2019. Visualizing and measuring the geometry of BERT. In *Proc. of NeurIPS*, pages 8592–8600, Vancouver, BC, Canada.

Kyle Richardson, Hai Hu, Lawrence Moss, and Ashish Sabharwal. 2020. Probing natural language inference models through semantic fragments. *Proc. of AAAI*, 34:8713–8721.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv:1910.01108 [cs]*. ArXiv: 1910.01108.

Tobias Schnabel, Igor Labutov, David Mimno, and Thorsten Joachims. 2015. Evaluation methods for unsupervised word embeddings. In *Proc. of EMNLP*, pages 298–307, Lisbon, Portugal. Association for Computational Linguistics.

Nathan Schneider, Jena D. Hwang, Vivek Srikumar, Jakob Prange, Austin Blodgett, Sarah R. Moeller, Aviram Stern, Adi Bitan, and Omri Abend. 2018. Comprehensive supersense disambiguation of English prepositions and possessives. In *Proc. of ACL*, pages 185–196, Melbourne, Australia. Association for Computational Linguistics.

Nathan Schneider and Noah A. Smith. 2015. A corpus and model integrating multiword expressions and supersenses. In *Proc. of NAACL-HLT*, pages 1537–1547, Denver, Colorado. Association for Computational Linguistics.

Harish Tayyar Madabushi, Laurence Romain, Dagmar Divjak, and Petar Milin. 2020. CxGBERT: BERT meets construction grammar. In *Proc. of ICCL*, pages 4020–4032, Barcelona, Spain (Online). International Committee on Computational Linguistics.

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 *Proc. of NeurIPS*, pages 5998–6008.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proc. of ICLR*. OpenReview.net.

Gregor Wiedemann, Steffen Remus, Avi Chawla, and Chris Biemann. 2019. Does BERT Make Any Sense? Interpretable Word Sense Disambiguation with Contextualized Embeddings. *arXiv:1909.10430 [cs]*. ArXiv: 1909.10430.

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 *Proc. of EMNLP*, pages 38–45, Online. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Proc. of NeurIPS*, pages 5754–5764, Vancouver, BC, Canada.

A Fine-tuning Setup

For the fine-tuning step, a simple linear projection layer is added to the end of the CWE model. HuggingFace's implementation of AdamW (Loshchilov and Hutter, 2019) is used as the optimizer with a learning rate of `2e-5` for 40 epochs. Prepositions in STREUSLE are annotated with two supersenses, but only the first one is used for fine-tuning.

B Full Results, Main Experiment

B.1 OntoNotes, $\ell < 500$, $r < 0.25$

Model	# FT Instances	Precision	Recall
random baseline	0	11.55	3.63
oracle	0	82.02	23.92
roberta-base	0	32.87	9.28
roberta-base	100	34.38	9.80
roberta-base	250	37.87	10.80
roberta-base	500	35.75	10.20
roberta-base	1000	36.88	10.48
roberta-base	2500	35.09	9.96
bert-base-cased	0	41.60	11.79
bert-base-cased	100	42.88	12.20
bert-base-cased	250	43.04	12.24
bert-base-cased	500	42.02	11.93
bert-base-cased	1000	43.42	12.37
bert-base-cased	2500	43.37	12.34
distilroberta-base	0	29.33	8.30
distilroberta-base	100	32.87	9.32
distilroberta-base	250	34.74	9.88
distilroberta-base	500	33.99	9.63
distilroberta-base	1000	32.24	9.16
distilroberta-base	2500	32.92	9.33
distilbert-base-cased	0	39.80	11.26
distilbert-base-cased	100	41.62	11.84
distilbert-base-cased	250	41.42	11.78
distilbert-base-cased	500	41.11	11.70
distilbert-base-cased	1000	40.70	11.55
distilbert-base-cased	2500	41.55	11.83
gpt2	0	18.34	5.37
gpt2	100	16.92	4.99
gpt2	250	18.08	5.30
gpt2	500	18.12	5.32
gpt2	1000	17.59	5.18
gpt2	2500	17.39	5.11
albert-base-v2	0	40.44	11.50
albert-base-v2	100	39.26	11.22
albert-base-v2	250	37.24	10.66
albert-base-v2	500	36.72	10.51
albert-base-v2	1000	38.01	10.86
albert-base-v2	2500	38.48	10.98
xlnet-base-cased	0	28.72	7.93
xlnet-base-cased	100	36.38	10.05
xlnet-base-cased	250	37.53	10.44
xlnet-base-cased	500	34.42	9.55
xlnet-base-cased	1000	37.38	10.36
xlnet-base-cased	2500	36.95	10.31

B.2 OntoNotes, $\ell < 500$, $r \geq 0.25$

Model	# FT Instances	Precision	Recall
random baseline	0	11.55	3.63
oracle	0	82.02	23.92
roberta-base	0	32.87	9.28
roberta-base	100	34.38	9.80
roberta-base	250	37.87	10.80
roberta-base	500	35.75	10.20
roberta-base	1000	36.88	10.48
roberta-base	2500	35.09	9.96
bert-base-cased	0	41.60	11.79
bert-base-cased	100	42.88	12.20
bert-base-cased	250	43.04	12.24
bert-base-cased	500	42.02	11.93
bert-base-cased	1000	43.42	12.37
bert-base-cased	2500	43.37	12.34
distilroberta-base	0	29.33	8.30
distilroberta-base	100	32.87	9.32
distilroberta-base	250	34.74	9.88
distilroberta-base	500	33.99	9.63
distilroberta-base	1000	32.24	9.16
distilroberta-base	2500	32.92	9.33
distilbert-base-cased	0	39.80	11.26
distilbert-base-cased	100	41.62	11.84
distilbert-base-cased	250	41.42	11.78
distilbert-base-cased	500	41.11	11.70
distilbert-base-cased	1000	40.70	11.55
distilbert-base-cased	2500	41.55	11.83
gpt2	0	18.34	5.37
gpt2	100	16.92	4.99
gpt2	250	18.08	5.30
gpt2	500	18.12	5.32
gpt2	1000	17.59	5.18
gpt2	2500	17.39	5.11
albert-base-v2	0	40.44	11.50
albert-base-v2	100	39.26	11.22
albert-base-v2	250	37.24	10.66
albert-base-v2	500	36.72	10.51
albert-base-v2	1000	38.01	10.86
albert-base-v2	2500	38.48	10.98
xlnet-base-cased	0	28.72	7.93
xlnet-base-cased	100	36.38	10.05
xlnet-base-cased	250	37.53	10.44
xlnet-base-cased	500	34.42	9.55
xlnet-base-cased	1000	37.38	10.36
xlnet-base-cased	2500	36.95	10.31

B.3 OntoNotes, $\ell \geq 500$, $r < 0.25$

Model	# FT Instances	Precision	Recall
random baseline	0	11.55	3.63
oracle	0	82.02	23.92
roberta-base	0	32.87	9.28
roberta-base	100	34.38	9.80
roberta-base	250	37.87	10.80
roberta-base	500	35.75	10.20
roberta-base	1000	36.88	10.48
roberta-base	2500	35.09	9.96
bert-base-cased	0	41.60	11.79
bert-base-cased	100	42.88	12.20
bert-base-cased	250	43.04	12.24
bert-base-cased	500	42.02	11.93
bert-base-cased	1000	43.42	12.37
bert-base-cased	2500	43.37	12.34
distilroberta-base	0	29.33	8.30
distilroberta-base	100	32.87	9.32
distilroberta-base	250	34.74	9.88
distilroberta-base	500	33.99	9.63
distilroberta-base	1000	32.24	9.16
distilroberta-base	2500	32.92	9.33
distilbert-base-cased	0	39.80	11.26
distilbert-base-cased	100	41.62	11.84
distilbert-base-cased	250	41.42	11.78
distilbert-base-cased	500	41.11	11.70
distilbert-base-cased	1000	40.70	11.55
distilbert-base-cased	2500	41.55	11.83
gpt2	0	18.34	5.37
gpt2	100	16.92	4.99
gpt2	250	18.08	5.30
gpt2	500	18.12	5.32
gpt2	1000	17.59	5.18
gpt2	2500	17.39	5.11
albert-base-v2	0	40.44	11.50
albert-base-v2	100	39.26	11.22
albert-base-v2	250	37.24	10.66
albert-base-v2	500	36.72	10.51
albert-base-v2	1000	38.01	10.86
albert-base-v2	2500	38.48	10.98
xlnet-base-cased	0	28.72	7.93
xlnet-base-cased	100	36.38	10.05
xlnet-base-cased	250	37.53	10.44
xlnet-base-cased	500	34.42	9.55
xlnet-base-cased	1000	37.38	10.36
xlnet-base-cased	2500	36.95	10.31

B.4 OntoNotes, $\ell \geq 500$, $r \geq 0.25$

Model	# FT Instances	Precision	Recall
random baseline	0	11.55	3.63
oracle	0	82.02	23.92
roberta-base	0	32.87	9.28
roberta-base	100	34.38	9.80
roberta-base	250	37.87	10.80
roberta-base	500	35.75	10.20
roberta-base	1000	36.88	10.48
roberta-base	2500	35.09	9.96
bert-base-cased	0	41.60	11.79
bert-base-cased	100	42.88	12.20
bert-base-cased	250	43.04	12.24
bert-base-cased	500	42.02	11.93
bert-base-cased	1000	43.42	12.37
bert-base-cased	2500	43.37	12.34
distilroberta-base	0	29.33	8.30
distilroberta-base	100	32.87	9.32
distilroberta-base	250	34.74	9.88
distilroberta-base	500	33.99	9.63
distilroberta-base	1000	32.24	9.16
distilroberta-base	2500	32.92	9.33
distilbert-base-cased	0	39.80	11.26
distilbert-base-cased	100	41.62	11.84
distilbert-base-cased	250	41.42	11.78
distilbert-base-cased	500	41.11	11.70
distilbert-base-cased	1000	40.70	11.55
distilbert-base-cased	2500	41.55	11.83
gpt2	0	18.34	5.37
gpt2	100	16.92	4.99
gpt2	250	18.08	5.30
gpt2	500	18.12	5.32
gpt2	1000	17.59	5.18
gpt2	2500	17.39	5.11
albert-base-v2	0	40.44	11.50
albert-base-v2	100	39.26	11.22
albert-base-v2	250	37.24	10.66
albert-base-v2	500	36.72	10.51
albert-base-v2	1000	38.01	10.86
albert-base-v2	2500	38.48	10.98
xlnet-base-cased	0	28.72	7.93
xlnet-base-cased	100	36.38	10.05
xlnet-base-cased	250	37.53	10.44
xlnet-base-cased	500	34.42	9.55
xlnet-base-cased	1000	37.38	10.36
xlnet-base-cased	2500	36.95	10.31

B.5 PDEP, $\ell < 500$, $r < 0.25$

Model	# FT Instances	Precision	Recall
random baseline	0	11.56	1.85
oracle	0	96.41	15.19
roberta-base	0	39.84	5.90
roberta-base	100	42.09	6.29
roberta-base	250	47.86	7.21
roberta-base	500	45.20	6.78
roberta-base	1000	48.99	7.37
roberta-base	2500	46.63	6.99
bert-base-cased	0	59.59	8.99
bert-base-cased	100	60.00	9.07
bert-base-cased	250	60.23	9.11
bert-base-cased	500	60.07	9.07
bert-base-cased	1000	59.94	9.07
bert-base-cased	2500	60.37	9.13
distilroberta-base	0	32.42	4.80
distilroberta-base	100	39.20	5.84
distilroberta-base	250	42.25	6.34
distilroberta-base	500	41.75	6.26
distilroberta-base	1000	40.09	6.02
distilroberta-base	2500	39.89	5.96
distilbert-base-cased	0	58.06	8.75
distilbert-base-cased	100	58.06	8.77
distilbert-base-cased	250	58.02	8.76
distilbert-base-cased	500	57.96	8.75
distilbert-base-cased	1000	58.03	8.75
distilbert-base-cased	2500	58.33	8.79
gpt2	0	21.75	3.24
gpt2	100	18.34	2.76
gpt2	250	21.53	3.17
gpt2	500	19.77	2.93
gpt2	1000	19.42	2.90
gpt2	2500	20.58	3.08
albert-base-v2	0	56.53	8.55
albert-base-v2	100	53.75	8.14
albert-base-v2	250	52.56	7.95
albert-base-v2	500	51.63	7.80
albert-base-v2	1000	51.97	7.86
albert-base-v2	2500	53.10	8.03
xlnet-base-cased	0	35.76	5.12
xlnet-base-cased	100	48.28	7.02
xlnet-base-cased	250	48.47	7.08
xlnet-base-cased	500	45.99	6.68
xlnet-base-cased	1000	48.80	7.13
xlnet-base-cased	2500	49.46	7.25

B.6 PDEP, $\ell < 500$, $r \geq 0.25$

Model	# FT Instances	Precision	Recall
random baseline	0	11.56	1.85
oracle	0	96.41	15.19
roberta-base	0	39.84	5.90
roberta-base	100	42.09	6.29
roberta-base	250	47.86	7.21
roberta-base	500	45.20	6.78
roberta-base	1000	48.99	7.37
roberta-base	2500	46.63	6.99
bert-base-cased	0	59.59	8.99
bert-base-cased	100	60.00	9.07
bert-base-cased	250	60.23	9.11
bert-base-cased	500	60.07	9.07
bert-base-cased	1000	59.94	9.07
bert-base-cased	2500	60.37	9.13
distilroberta-base	0	32.42	4.80
distilroberta-base	100	39.20	5.84
distilroberta-base	250	42.25	6.34
distilroberta-base	500	41.75	6.26
distilroberta-base	1000	40.09	6.02
distilroberta-base	2500	39.89	5.96
distilbert-base-cased	0	58.06	8.75
distilbert-base-cased	100	58.06	8.77
distilbert-base-cased	250	58.02	8.76
distilbert-base-cased	500	57.96	8.75
distilbert-base-cased	1000	58.03	8.75
distilbert-base-cased	2500	58.33	8.79
gpt2	0	21.75	3.24
gpt2	100	18.34	2.76
gpt2	250	21.53	3.17
gpt2	500	19.77	2.93
gpt2	1000	19.42	2.90
gpt2	2500	20.58	3.08
albert-base-v2	0	56.53	8.55
albert-base-v2	100	53.75	8.14
albert-base-v2	250	52.56	7.95
albert-base-v2	500	51.63	7.80
albert-base-v2	1000	51.97	7.86
albert-base-v2	2500	53.10	8.03
xlnet-base-cased	0	35.76	5.12
xlnet-base-cased	100	48.28	7.02
xlnet-base-cased	250	48.47	7.08
xlnet-base-cased	500	45.99	6.68
xlnet-base-cased	1000	48.80	7.13
xlnet-base-cased	2500	49.46	7.25

B.7 PDEP, $\ell \geq 500$, $r < 0.25$

Model	# FT Instances	Precision	Recall
random baseline	0	11.56	1.85
oracle	0	96.41	15.19
roberta-base	0	39.84	5.90
roberta-base	100	42.09	6.29
roberta-base	250	47.86	7.21
roberta-base	500	45.20	6.78
roberta-base	1000	48.99	7.37
roberta-base	2500	46.63	6.99
bert-base-cased	0	59.59	8.99
bert-base-cased	100	60.00	9.07
bert-base-cased	250	60.23	9.11
bert-base-cased	500	60.07	9.07
bert-base-cased	1000	59.94	9.07
bert-base-cased	2500	60.37	9.13
distilroberta-base	0	32.42	4.80
distilroberta-base	100	39.20	5.84
distilroberta-base	250	42.25	6.34
distilroberta-base	500	41.75	6.26
distilroberta-base	1000	40.09	6.02
distilroberta-base	2500	39.89	5.96
distilbert-base-cased	0	58.06	8.75
distilbert-base-cased	100	58.06	8.77
distilbert-base-cased	250	58.02	8.76
distilbert-base-cased	500	57.96	8.75
distilbert-base-cased	1000	58.03	8.75
distilbert-base-cased	2500	58.33	8.79
gpt2	0	21.75	3.24
gpt2	100	18.34	2.76
gpt2	250	21.53	3.17
gpt2	500	19.77	2.93
gpt2	1000	19.42	2.90
gpt2	2500	20.58	3.08
albert-base-v2	0	56.53	8.55
albert-base-v2	100	53.75	8.14
albert-base-v2	250	52.56	7.95
albert-base-v2	500	51.63	7.80
albert-base-v2	1000	51.97	7.86
albert-base-v2	2500	53.10	8.03
xlnet-base-cased	0	35.76	5.12
xlnet-base-cased	100	48.28	7.02
xlnet-base-cased	250	48.47	7.08
xlnet-base-cased	500	45.99	6.68
xlnet-base-cased	1000	48.80	7.13
xlnet-base-cased	2500	49.46	7.25

B.8 PDEP, $\ell \geq 500$, $r \geq 0.25$

Model	# FT Instances	Precision	Recall
random baseline	0	11.56	1.85
oracle	0	96.41	15.19
roberta-base	0	39.84	5.90
roberta-base	100	42.09	6.29
roberta-base	250	47.86	7.21
roberta-base	500	45.20	6.78
roberta-base	1000	48.99	7.37
roberta-base	2500	46.63	6.99
bert-base-cased	0	59.59	8.99
bert-base-cased	100	60.00	9.07
bert-base-cased	250	60.23	9.11
bert-base-cased	500	60.07	9.07
bert-base-cased	1000	59.94	9.07
bert-base-cased	2500	60.37	9.13
distilroberta-base	0	32.42	4.80
distilroberta-base	100	39.20	5.84
distilroberta-base	250	42.25	6.34
distilroberta-base	500	41.75	6.26
distilroberta-base	1000	40.09	6.02
distilroberta-base	2500	39.89	5.96
distilbert-base-cased	0	58.06	8.75
distilbert-base-cased	100	58.06	8.77
distilbert-base-cased	250	58.02	8.76
distilbert-base-cased	500	57.96	8.75
distilbert-base-cased	1000	58.03	8.75
distilbert-base-cased	2500	58.33	8.79
gpt2	0	21.75	3.24
gpt2	100	18.34	2.76
gpt2	250	21.53	3.17
gpt2	500	19.77	2.93
gpt2	1000	19.42	2.90
gpt2	2500	20.58	3.08
albert-base-v2	0	56.53	8.55
albert-base-v2	100	53.75	8.14
albert-base-v2	250	52.56	7.95
albert-base-v2	500	51.63	7.80
albert-base-v2	1000	51.97	7.86
albert-base-v2	2500	53.10	8.03
xlnet-base-cased	0	35.76	5.12
xlnet-base-cased	100	48.28	7.02
xlnet-base-cased	250	48.47	7.08
xlnet-base-cased	500	45.99	6.68
xlnet-base-cased	1000	48.80	7.13
xlnet-base-cased	2500	49.46	7.25