BERTological Lexicography for Prepositions

Ken Litkowski
CL Research
9208 Gue Road
Damascus, MD 20872 USA
ken@clres.com

December 18, 2022

Abstract

Contextualized word embedding (CWE) models such as BERT (Devlin et al. (2019)) have been used in many NLP tasks. Gessler and Schneider (2021) (G&S) focus on the use of BERT in disambiguating rare senses of nouns, verbs, and prepositions. In examining preposition disambiguation, their results are comparable to earlier efforts and achieving better results than problems described in Litkowski (2013). Earlier efforts achieved results using traditional NLP methods, characterizing syntactic and semantic behaviors. BERTology provides a new resource and additional perspective that might assist in usual lexicographic procedures. We begin exactly at the place where G&S ended, using its methods, adding the further property of keeping a link to the instances in the Pattern Dictionary of English Prepositions (PDEP, Litkowski (2014)), enabling further lexicographic analysis. Our paper provides a lexicological analysis of the instances, which may provide the basis for lexicography.[1]

1 Introduction

Contextualized word embedding (CWE) models such as BERT (Devlin et al. (2019)) have been used in many NLP tasks. Gessler and Schneider (2021) (G&S) discusses these tasks and particular indicates that "it is not easy to understand exactly which aspects of linguistic form and meaning contextualized word embeddings are able to capture." While they focus on the

[1]Working Paper 22-02. Damascus, MD: CL Research. All code is available at `https://github.com/kenclr/BERTLex`.

use of BERT in disambiguating rare senses, their methods provided implementations that provide further investigation of lexicography. We have implemented their (Python) methods and initiated further lexicographic examinations.

Lexicography goes beyond disambiguation. A dictionary defines the communicative and cognitive functions of a word and chooses the structures for presenting the words, such as a frame and distribution structure. A lexicographer will examine additional information characterizing a word. CWE models provide a huge amount of data that can benefit a lexicographer. G&S provides a starting point. We use the preposition material as in G&S, from the corpora in the Pattern Dictionary of English Prepositions (PDEP, Litkowski (2014)).

In section 2, we describe the relevant material provide the general approach to synchronizing the two compilations (PDEP and BERT). In section 3, we describe the three PDEP corpora, particularly indicating how they differ from each and why it may be useful to include the metadata for each instance. the steps using the tagging and the initial output of this process. Section 4 shows the distribution of the three corpora for the PDEP instances, particularly moving the emphasis from multiple-word to single-word prepositions, describing these changes in section 4.1. In section 5, we characterize the groups of PDEP senses, one used in the G&S analysis and four other groups not subjected to further detailed analysis. Section 6 provides several lexicographic analyses of the 346 test instances that were predicted in the G&S paper. Section 7 identifies several other questions that can be addressed and characterized at BERTological lexicography. Section 8 identifies other papers that have aspects of BERTology that might be useful for lexicography.

2 The Contour for Word Sense BERTology

G&S uses two corpora, one from OntoNotes 5.0 (Hovy et al. (2006)) for nouns and verbs and one from the Pattern Dictionary of English Prepositions (PDEP, Litkowski (2014)) which we will use here. The PDEP corpus consists of 81,489 sentences. G&S used this corpus to construct a test set and a training set used to predict the test instances for the most common 48 prepositions, discarding infrequent senses (fewer than 5 times). The training set contained 33,090 instances and the test set contained 8,020 instances.[2]

[2]This discussion contains only a small amount of G&S. More of the details can be seen the paper (Gessler and Schneider (2021)). As described there, all code for that

The key methodology uses BERT[3] to characterizes each test and train sentence. Each test sentence is encoded (e.g., *'Pry now asks you to advise him >>about<< a house he wants to buy .'*, focusing on the target preposition. The object is to predict which sentences of the training set are closest to the test sentence, using the cosine metric. In predicting, the subset of the training set uses sentences with the target preposition. Where possible, the 50 nearest sentences are identified. For example, the closest training sentence to the above test is *'The hostel staff advised us >>about<< the best areas for walking ...'*.

The predictions are printed into a tab-separated value table, with a row for each test instance. Each row contains 204 cells, consisting of the test sentence, the label (sense) of the preposition, the lemma (i.e., without the sense identifier), the label frequency for the sense in the training set, and the results of the 50 closest training items. Each closest item has four pieces of information: the label of the training sentence, its lemma, its sentence, and the distance of the training sentence to the test sentence. In summary, each test is characterized with 204 cells of data. This table is saved into a Python DataFrame where it can be used to examine the results.[4]

We made a few modifications and additions to the code. The G&S code provided with the ability to create a different random shuffling to the PDEP corpus.[5] Rather than doing so, we used the shuffled version that is available[6] so that we could replicate G&S. Also, in predicting, the test instances with the same label are shuffled in batching[7]. This does not seem to be significant for the PDEP instances, but would result only in modifying the ordering of the test set. We did not use this shuffling, again with the idea of replicating G&S.

In reading the PDEP instances,[8], several fields are constructed each sentence: a TextField (for the tokens in a sentence), a SpanField (identifying

---

work is available at `https://github.com/lgessler/bert-has-uncommon-sense`. We have implemented their Python code locally as the starting point for the discussion in this paper.

[3]bert-base-cased

[4]This table, pdep-bert-base-cased_7.tsv, is included the **Data** at `http://www.clres.com/online-papers/BERTLex.zip`. Details of changes to G&S will be provided below if others wish to make even further changes.

[5]Using bhus/scripts/format_pdep.py. See changes to this script in our implementation. The script bhus/scripts/counting.py characterizes the PDEP instances used and not used in G&S; these counts are described in sections 3, 4, and 5 and included in files in the folder cache/clres_stats.

[6]bhus/data/pdep/pdep_test.conllu and bhus/data/pdep/pdep_train.conllu

[7]In the function batch_queries in bhus/common/util.py.

[8]In ClresConlluReader in bssp/clres/dataset_reader.py

the location of the preposition in the sentence), a LabelField (identifying the sense number of the preposition), a LemmaField (the lemma for the preposition), and an ArrayField (characterizing embeddings for the preposition). In constructing these fields, each sentence assured that the source of the sentence was acceptable, giving the corpus name, the instance number of the corpus, and the location of the preposition (i.e., the one that used for the SpanField). This meta data was not further used in G&S; we added this meta data, created a MetadataField. This field is also used below in the the lexicographic analysis making use of the BERT data.

3 The Three PDEP Corpora

PDEP has three corpora: OEC, FN, CPA. The tagging of these corpora may be viewed in this order in their accuracy. OEC were generated through the most amount of effort, with professional lexicographers and leading to entries in the Oxford Dictionary of English (ODE, Stevenson and Soanes (2003)). The entries were supported with up to 20 examples from the OEC, curated by the lexicographers. FN instances were tagged by a single professional lexicographer, using the sense inventory from the ODE, but also with some additions based on judgment. These instances cover only 57 prepositions, out of the total 303, not including multiple-word prepositions. The FN instances were generated based on criteria that were not intended to be representative. CPA instances were tagged by a computational lexicologist, trying to be conforming with the OEC and FN tagging. These instances were intended to be representative, drawn from the British National Corpus.

As indicated above (section 2), G&S selected a subset (41,110) of the PDEP instances (81,489) for its analyses. The BERT embeddings for the selected subset appear to provide a considerable level of characterization, to the extent may be very useful in lexicography. Adding the metadata for each instance will allow investigation of the differences between the three corpora. In particular, these sense tags raise several questions worth analysis and examination.[9]

[9]See the functions **pdepinsts** and **unused** in **counting.py**. There are actually 82,329 instances in the source, but 840 instances were not included in the online PDEP. These instances ('v', 'up until', 'in connexion with', 'apropos of', 'vs.', 'in course of', 'amongst', "'fore", 'on the heels of') were not included.

Data	**FN**	**CPA**	**OEC**	**Total**
PDEP Instances	26739	47285	7465	81489
No sense	160	6480	0	6640
Idioms	498	19975	2006	22479
Not located	68	1	1	70
Uncommon	2833	7392	1552	11777
G&S Instances	23180	13437	3906	40523

Table 1: Eliminating Instances from Analysis

4 Identifying the Analysands

Tables 1 and 2 show the distribution of the three corpora for the PDEP instances, particularly showing that the CPA corpus had 58 percent of the instances. Since the CPA corpus was developed as representative from the British National Corpus (see Litkowski (2014)), many of the putative prepositions are actually adverbs or parts of phrasal verbs and needed to be eliminated from the analysis. G&S screened each instance to assess suitability for the BERTology analysis. They removed any super-rare instances that did not occur at least 5 times in the train set, viewing them as not sufficient for evaluation.

Table 1 shows the four major criteria used to exclude instances. The first criterion was exclude instances that had no preposition sense: an empty sense, instances not words that were prepositions (usually adverbs), and words that were part of phrasal verbs (e.g., *come across*). The next set of exclusions was for instances that corresponded to phrasal prepositions, containing a space to indicate idioms (multiple word expressions, MWEs). Several instances have some problem locating the preposition (e.g., offset issues). Finally, instances viewed as uncommon were eliminated. The script listed a WHITELIST of 48 acceptable prepositions, eliminating 78 single-word prepositions (see Appendix A, noting that 29 prepositions perhaps could have been included as common). As indicated in Table 2, the relative instances change dramatically, with FN and CPA switching percentages. These percentages will be relevant in the below discussions.

Tables 3 and 4 show the absolute and the relative predictions of the data frame. The first line indicates how many instances were used in making the predictions. The second line shows the 8,231 test instance in each corpus. The two bottom lines in these tables show the nearest prediction

Data	**FN**	**CPA**	**OEC**
PDEP Instances	0.328	0.580	0.092
G&S Instances	0.572	0.332	0.096

Table 2: Relative Instances in Analysis

Data	**FN**	**CPA**	**OEC**	**Total**
Train Instances	18495	11467	3128	33090
Test Instances	4673	2773	785	8231
Prediction Instances	4995	2372	864	8231
Correct Predictions	4169	1767	686	6622

Table 3: Instances in the Initial Dataframe

instances and whether they correspond to the sense in the test instance.[10] The number of predictions is equal to the number of test instances, but they differ, indicating that FN and OEC are more likely to use them as the first prediction, with CPA using less frequent. Table 4 shows the percent of the predictions were correct, indicating that FN was much more likely to be correct than the other two corpora.

4.1 Relative Accuracy of Corpora Instances

Table 4 suggests that using BERT achieves excellent results, even using only the first (of the 50) prediction. We next examine the extent to which this first prediction uses the same corpus as the corpus in the test instance.[11] As shown in Tables 5 and 6, each of the corpora has the largest number and

[10]In **ranks.py**, functions **corpanal** and **corp2**.
[11]In **ranks.py**, function **corp_test**.

Data	**FN**	**CPA**	**OEC**
Train Instances	0.559	0.347	0.095
Test Instances	0.568	0.337	0.095
Prediction Instances	0.607	0.288	0.105
Correct Predictions	0.506	0.215	0.083
Percent Correct	0.834	0.747	0.790

Table 4: Percent in the Initial Dataframe

Test Corpus	**FN**	**CPA**	**OEC**	**Total**
FN Instances	4146	364	163	4673
CPA Instances	667	1844	262	2773
OEC Instances	182	164	439	785
Total	4995	2372	864	8231

Table 5: Correspondence Between Test and Prediction Corpus

Test Corpus	**FN**	**CPA**	**OEC**
FN Instances	0.887	0.078	0.035
CPA Instances	0.241	0.665	0.094
OEC Instances	0.232	0.209	0.559
Total	0.607	0.288	0.105

Table 6: Percentage Correspondence Between Test and Prediction Corpus

percentages of the same corpus. FN is the most dominant use of itself (at 88.7%), also predicting substantial chunks for the test instances of the other two corpora, so that FN is a prediction of 60.7% for the totality of the three corpora.

These results may give a misleading impression, with different perceptions for the predictions of the individual prepositions. From the initial 48 prepositions, removing the eight prepositions (B) with only one sense leaves 40 polysemous prepositions.[12]

5 PDEP Senses

As indicated above, PDEP has 1039 senses, most having sentences that exemplify them. Below, we describe the 207 senses without any instances (section 5.1) and the 832 senses having instances (section 5.2).[13] We provide details about these groups, particularly to compare the senses used in the G&S paper. This comparison is not done with the idea of criticizing the paper, but rather to characterize the kinds of issues to be considered in BERTological lexicography.

[12]In **ranks.py** with the function **prep_anal**.

[13]In **counting.py** with the functions **defs**, **with_insts**, and **no_insts**

5.1 Senses Without Instances

PDEP has 207 senses that have no instances in the three corpora.[14] Most (180) of these senses come about from the treatment of the 24 variants in PDEP. These variants are characterized as being in the **Tributary** class. The senses for these variants repeated the sense inventories for the base preposition. For example, *frae* uses the 16 senses of *from* and then adds another sense indicating characterizing it as a variant for the base preposition. Each of the variants had one or a few instances, but did not have instances for many of the senses from the base preposition.

The remaining 27 senses without instances correspond to rare or special treatment. The first senses for *by* and *of* are senses that have two subsenses, both having very large numbers of instances, but not with the supersense. Others, such as *by (10(2e)): "indicating the sire of a pedigree animal, especially a horse"*, had no instances. Others were senses that included the ODE inventories (such as *gone* and *vice*) but without any example instances.

5.2 Senses With Instances

PDEP has 832 senses with instances in one of the corpora. G&S predicts test instances covering 346 senses, but 11 of these senses are not in PDEP. The lexicographer tagging the FN corpus used some "senses" containing 2 or 3 PDEP senses, i.e., viewing as polysemous. Notwithstanding the multiple senses, there were enough instances that they were predicted in the G&S analysis. Later, we describe the G&S senses below in section 6. In this section, we describe the 497 senses (832 - 335) that were not analyzed in the G&S paper.[15] There are three groups:

- **Multi-Word Expressions (MWEs)**: PDEP contains 269 senses in 165 entries, such as *because of* and *along with*.[16] The number of senses and entries indicates that some MWE entries are polysemous. G&S excluded instances having MWEs. While many of these prepositions have only one sense, MWE disambiguation is a subject for further analysis. Since three of the senses in two entries have no instances in PDEP[17], there are only 266 senses in 163 entries in counting in this subsection.

[14]In counting.py with the function **no_insts**.

[15]In **counting.py** with functions **csenses** (867) minus **nondefs** (35), minus **senses** from **ranks.py** (346) minus **nondefs** (11) in **counting.py**.

[16]In **counting.py** with **mwe** function.

[17]*back of_1(1)*, *in behalf of_2(2)*, and *in memoriam_1(1)*

- **Uncommon Single-Word Prepositions**: G&S established a whitelist of prepositions to be used for the analysis. As a result, 78 single-word prepositions with 181 senses (see appendix A) not in this list were treated as "uncommon".[18] In reviewing this list, as many as 29 prepositions could be viewed as "common", i.e., polysemous and having a sufficient number of instances supporting disambiguation and prediction. This includes *by*, *throughout*, and *within*. This is not problematic with regard to the G&S analysis, since each preposition can be analyzed and predicted independent of what is happening for results of other prepositions.

- **Senses of Common Words Not Predicted**: G&S includes 48 prepositions in its whitelist, but 8 of these are monosemous (see B). Of the remaining, 26 of the 40 polysemous prepositions have 50 senses that were not used in the analysis because they had very few instances (see B), i.e., insufficient for including in the predictions. Notwithstanding, these senses have a few instances that are included for considering in the distance analysis. For example, *on_21(11)* (*paid for by*, e.g. *dinner's on me*) does not have any instances among the test set, but several of the instances in the train set were identified as being among the nearest 50 instances.

6 Overview of Sense Predictions

G&S describes one PDEP sense (in its Figure 1), with further analysis in its Results (section 5). This is for sense 20 of *on*, "regularly taking (a drug or medicine)". They suggest that "BERT is prioritizing syntactic criteria ('on' following a verb, especially a copula)", where this characterization is based on general linguistic analysis rather than based on BERTological analysis. This discussion can be viewed as appropriately lexicographic.

G&S predicted results for 8231 test instances. In this section, we provide further examinations of all the test instances, following the procedure that G&S used for the one sense. The 8231 instances covered 346 senses, an average of about 24 instances for each sense. They envisioned that top 50 predictions would be computed. Fewer than 50 results would be generated only when a preposition had fewer than 50 instances in the training instances for a lemma, i.e., all senses, not just the sense of the test instance. This occurs only for *circa*, which has 3 test queries and 17 training instances.

[18]In **counting.py** with the function **uncommon**.

G&S also has 11 senses that are not in PDEP (B). Predictions were made in 60 test instances for these senses. The tags for these senses were used by the FN lexicographer. One sense (*onto_3(3)*), with one test query, was an error. The other 10 senses were tagged by the FN lexicographer as having two senses, either from ambiguity or judging that both were applicable. Having multiple senses is not problematic, but may rather be suggestive lexicography. [19]

6.1 Procedures for Analysis of Predictions

In describing sense 20 of *on*, using one of the five test queries, G&S indicated that there were only 14 occurrences of this sense in the training set. They showed the nearest six results, with the first four using the same sense, and with only six of the 14 instances made it into the top 50 for the query test. Our objective was to examine these predictions for each of the 8231 test queries.[20]

The predictions are contained in a tab-separated table, into a dataframe which is used for the analyses. The results consist of 204 columns for each test query: the test sentence, the label (i.e., the preposition sense, along with corpus instance number for the sentence), the lemma, the number of training sentences with the label, and four items for the 50 nearest instances (the prediction and the corpus instance number for the prediction sentence, its sentence, its lemma, and the distance). The table is the starting point for examinations. Importantly, each prediction is independent of each other, either for test queries using the same sense, other senses of the same lemma, or other lemmas. As this discussion proceeds below, further questions will arise for other analyses.

G&S described only one test query, one that has fewer than 51 training instances (here 14), so that the top 50 predictions must have some other senses of the lemma. This situation leads to two groups of the 346 senses, with 161 senses having more than 50 training instances (high frequency senses) and 184 senses having 50 or fewer instances (low frequency senses).[21] For the low frequent senses, the question is whether all training instances

[19]For example, "He studied **at** Ealing School of Art", *1(1)*: expressing location in a particular place, and/or *4(2b)*: denoting the time spent by someone attending an educational institution.

[20]The data for these analyses are contained in three tab-separated files: **predictions**, **ranks**, and **scores** and available online, where they can be examined by anyone else. The Python scripts will be included as well. The full script is contained in **ranks.py**; we import all the functions in this file.

[21]In **ranks.py** with the function **dfsanal**.

occur in the top 50 predictions, and if not, how many have not been found. For the high frequent senses, the question is whether all 50 top predictions have the same sense as the test query, and if not, how many predictions used other senses for the lemma.

In the top 50 predictions, it is important to recognize that a polysemous preposition could have several senses with no test but still be the predicted sense. Also, when training instances have a very small number of instances in the test set and a relatively just a few more such instances in the train set could still be a predicted sense.

6.2 Nearest Training Instance

The first question asks whether the nearest training instance matches the test query sense.[22] To answer this question, we iterate through the rows of the prediction table, counting the number of senses having more or fewer 50 number of training instances, incrementing the count of matches when the nearest training instance (*label_1*) has the same *label* as the test query.

For the 159 high frequent senses, there are 7053 test queries; of these, the two labels are the same in 5842 cases, at 0.828. For the 187 low frequent senses, there are 1178 test queries; of these, the nearest training instance has the same labels in 780 cases, at 0.662. These results suggest that the most frequent senses are easier to match and the least frequent senses are much more difficult to match. However, these results might be an artifact (e.g., see Litkowski (2013)) since the PDEP instances used in G&S are not representative of the general corpus. Notwithstanding, the use of the BERTological data provides several mechanisms for further analysis. For example, we can ask the question of whether simply a large number of examples would improve examination of the nearest training instances.

For 52 test queries, the distance of the nearest prediction is 0.0. There are several reasons for this situation, arising in the corpora.[23] Despite the small number of these cases, they provide appropriate lexicographical descriptions that need for consideration. Most of these cases (45) have the same sense; in the remaining instances, the prediction is different from the sense of the test query.

For the cases with the same senses for the test query and the first prediction, there are four types:

[22]This uses the function **nearest(df)** in **ranks.py**, modifying it for either more or fewer 50, with *testlab* as the number in the test queries (*label*) and *near* as the number matching the sense of the test query (*label_1)*.

[23]In **ranks.py** with the function **score_low**,

- In the FN corpus, 26 of the test queries have duplicates in the corpus. This occurs because a sentence was selected for two different prepositions and were incorrectly tagged based on the same preposition.
- In the FN corpus, 4 of the test queries have duplicates but only one preposition. There was no explanation for the duplications.
- In the CPA corpus, there were two instances with the same sentence. There was no explanation for the duplications.
- For 13 test queries, the same sentence occurred in the FN and the CPA corpora. This occurred because these two corpora are both used as the same original corpus.

For the instances where the prepositions were tagged with two different senses, there are of two types:

- Five cases have different taggings, indicating disagreement in the tagging,
- In the OEC corpus, the same instances were used for two senses of *for*, for *4(3a), employed by* and *5(4), having (the thing mentioned) as a purpose or function.*

These situations are the kinds of detail that are relevant to any BERTological analysis.

6.3 Ranking the Predictions

To continue the description used in G&S, we further characterized the predictions.[24] We iterated through the 8231 test queries making several counters. A rank for each test query assesses how well the predictions correspond with the training instances having the same label (senses) as the test query. First, we counted the number of test queries for each sense (*label*), i.e., incrementing the counter by one. Second, we added the number of training instances for each sense, incrementing the counter by this frequency for each sense, i.e., eventually multiplying the frequency by the number of test queries. When the frequency was greater than 50, we reset the frequency to 50 (corresponding to the top 50 predictions).

We created an empty list (*ranks*) to hold the positions for the places where a *label_n* equaled the *label.* We iterated through the *i* over the range

[24]This uses the function **rank** in **ranks.py**, generating two dataframes, one for **ranks** in 6.3.1 and one for **scores** in 6.3.2.

from 1 to 50. When equal, we added that rank to *ranks*. If i was less than the training frequent, we incremented a counter at *score* by the frequency minus 1. A score of 50 was given if the first prediction occurred as the nearest instance, and so down to 1 for *label_50*. At this point, the length of *ranks* indicated how many of the 50 predictions had the same as *label*. We incremented a counter at *found* by this length, so that this would show how many matched for all of the test queries for a given sense. Similarly, we incremented a counter at *scores*. The total score for a test query could have a maximum of $\mathbf{n(n+1)/2}$, i.e., the inverse of the sum of the integers from 1 to n. This maximum was incremented to a counter at the optimum (*opts*) possibility.

6.3.1 The Ranks Dataframe

At this point, we had the data to characterize the performance each test query, each adding to a row in a dataframe. This row consisted of (1) the label (sense), (2) the number of training instances for the sense (the same value for each row for the same label), (3) how many of the training instances were matched, (4) the score, (5) the optimum score possibility, (6) the full label (the same as the label, but also containing the corpus source and instance number), and (7) the full number of the training frequency (in cases where this was more than 50). We will make some observations about these below.

6.3.2 The Scores Dataframe

The next step is to summarize the test queries for each of the 346 labels (senses).[25] We iterate through the senses making some calculations into a dataframe row for the *scores*. We set *tasks* to the number of test queries for each sense. We set *in50* to *found[i]/freqs[i]*, the number of training instances that were found divided by the desired frequencies. We set *pct* to *scores[i]/opts[i]*, the scores divided by the optimum scores, i.e., the overall percentage for the sense. A row consisted of (1) the label (sense), (2) the number of tasks for the sense, (3) the label frequency in the training set, (4) a string showing *in50*, (5) the coverage (the percentage found in top 50 predictions), (6) a string showing the scores over the optimums, and (7) the overall optimum score for the sense.

[25]This is not included for the preposition *circa*, which had 3 instances in the test set and 17 in the train set. This is the only lemma that doesn't have 50 predictions.

6.4 Observations About the Ranks

When the label frequency is equal to or greater than 50 (reset to 50), the optimum score is 1275. If the score is less than the optimum, this means that a sense other than the label in the test query has been predicted. When the label frequency is less than 50, we know that other senses for the lemma than the one in the test query. We can scroll through each test query and see ones that are significantly different, suggesting that the test query may be incorrectly tagged. It is possible that the test query is ambiguous and might be potentially tagged with more than one sense.

Scrolling through the dataframe for the test queries, distinctive instances jump out. For example, there are 236 test queries for *about_1(1)*, with 962 training instances. With this many training instances, most of the 50 predictions have the same sense, with only a few other senses for *about*. Thus, it is striking when the score is significantly lower than 1275. We can quickly see the sentence that resulted the distinctive difference.[26] We can see how many of the training instances were found in the top 50 predictions, and how closely the found instances were the top predictions. When there are several test queries, we may see that there may be grouping into sets, perhaps wondering if there ought to be different definitions.

6.5 Observations About the Scores

As indicated above (6.3.2), the scores dataframe summarizes the results for all tasks with the same label (sense). For each sense, we can see the number of tasks and the number of instances (sentences) in the training set for that sense. These two numbers are multiplied to show the number of occurrences as the denominator to evaluate for the sense. In the *in50* column, we show how many of the top 50 predictions are present. This number is divided by the denominator to indicate the **coverage** (*cover* column), ranging from 0 to 1. This column can be sorted and can be viewed to get an idea of which senses for a preposition are more likely to have the variation of coverage. In the *score* column, we show, for all the senses' test queries, the total scores out of the optimal corresponding the ranks of each prediction. This score is shown in the **opt** (*optimal* column), also ranging from 0 to 1.

The first analysis of the dataframe examines the difference between 161 senses with high training frequencies (≥ 50) and 184 senses with low training frequencies (<50).[27] The average coverage for the high frequency senses is

[26] `https://www.clres.com/db/TPPEditor.html`

[27] The function **dfsanal** in *ranks.py*.

0.62 and for the low frequency senses is 0.52. The average optimality for the high frequency senses is 0.66 and for the low frequency senses is 0.469. In general, the predictions for the high frequency senses are more likely to correspond to the sense of the test query, whereas the predictions for the low frequency senses are more likely to use other senses of the preposition than the sense of the test query. Similarly, the predictions for the low frequency test queries are less likely to be the nearest senses.

There are 21 senses that have coverage equal to 1 (sorting the *cover* column); these include the 8 prepositions that have only one sense (Appendix B), i.e., they cannot be polysemous. The others have a very few test queries (1, 2, 3, 5, or 9), with a small number of training instances (6 to 16). Thus, it doesn't take much for all the training instances to occur in the top 50 predictions.

Senses with high coverage seem to occur in two situation. If a sense is very distinctive among a preposition's inventory, the instances for the training set will tend to be quite similar. Another situation is for the first sense of the preposition's inventory. For example, *about_1(1)* has 236 test queries out of a total of 264 (89 percent), so that it is quite likely to be selected and thus has high coverage and very close to the optimal, i.e., seldom selecting a different sense.

When the coverage and/or optimum is medium (e.g., 0.50 ± 0.20), the suggestion is that the sense is not distinctive. This occurs when the preposition has several senses that are close to each other. In these cases, the training instances do not distinguish among the similar senses, identifying nuances of meaning, captured by the sense inventory.

When the coverage and/or optimum is low, the training instances are clearly inadequate to characterize the test query sense. In these cases, use of the PDEP instances are appropriate in an attempt to understand why these results are occurring. Examining the CPA instances in comparison with the OEC and the FN instances may indicate that the PDEP tagging may have been incorrect. Future steps in retagging training instances are discussed below (section 7.3). OEC instances may also use only a few forms, but may occur in several other forms that will not be predicted. For example, *through_9(2b)* (*from beginning to end of (an experience or activity, typically a tedious or stressful one)* primarily uses *go* and *been* modified by this sense, but may use several other verb forms in the CPA and FN instances, with the result of having lower coverage and optimality. The very next sense, *through_10(3)* (*so as to inspect all or part of (a collection, inventory, or publication)* uses only a few verbs (such a *search through* or *skim through*, while the OEC instances with *go through* or *hunt through*; such training instances make it difficult to predict the test query.

Each test query provides considerable interest in its predictions. With the 345 test queries (and other that could have been sufficient training numbers), it is likely that generalizations might be possible with further examinations. This is the essence of lexicography that can use BERTology methods that have been described above.

6.6 Frequency and Centrality of Training Instances

In considering the top 50 predictions for each test query, we wondered whether the nearest neighbors clustered, i.e., some appeared to be more important for the several instances with the same sense. With the metadata incorporating into the predictions, identifying the corpus and sentence instance number, frequency and centrality of the training sets can be examined. We begin with the scores dataframe to use the number of task queries (column *tasks*) and training instances for a sense (label) (column *freq*). We use three counters for the metadata item, using the concatenation of the corpus name and instance number.[28]

The first counter simply increments the number of occurrences of the corpus-instance in the test queries.[29] This number is the numerator of the coverage calculation in the scores. For *against_9(3a)* (*in visual contrast to*), there are 21 instances, with 5 test queries and 16 training instances. When we examine the top 50 predictions, a corpus-instance may occur in all test queries . In this case, we have 16 training instances and all of them occurred in the predictions, so that when we sum this column we obtain 80 (5 * 16) of the column *occ*.[30] For *for_4(3a)*, we have 9 tests; we have 48 training instances. However, only 46 training instances occur and none of these instances occur in all tests; the sum of this column is 209, much lower than the full occurrences (432 = 9 * 48).[31] For *at_1(1)*, we have 112 tasks with 479 training instances; with only 50 possible predictions, the maximum is 5600 (112 * 5).[32] In this case, the sum of this column is 4750; the column contains 462 corpus-instances, with the most occurs only 53 of the 112 test queries,

The second counter increments the position of the corpus-instance in the 50 predictions for the sense (the column *locs*). If a training instance is the

[28]The function **avescore** in **ranks.py**.

[29]In **dfscores**, columns *tasks* and *freq*

[30]**against['occ'].sum()**

[31]**for['occ'].sum()**

[32]**at['occ'].sum()**

nearest neighbor for all the test queries, i.e., has position 1, the sum of the positions would equal to the number of test queries. This is highly unlikely to occur. For the 16 corpus-instances of *against_9(3a)*, the sums range from 21 to 99. The instance with the smallest sum can be viewed as most similar to the combination of all the test queries. The range for the sums appear to show an increasing variation and divergences from the test queries.

The third counter increments the distance of the corpus-instance in the 50 predictions for the sense (the column *dists*. Since the test queries for a sense are independent of each other, the distances are not essentially comparable to one another. For example, *against_9(3a)* has three test queries with the same locations (at 27), but their distances are different. In general, all average distances for the predictions will need to be examined in detail. These distances do provide useful data for further analysis.

Examining the dataframes for the senses involves several properties. We first computed the "average" position and the average distance for each training instance. Only 20 of the 346 have predictions with the test query corresponding to the training frequencies. In general, a corpus instance with a prediction that matches the test query will not occur in all the test queries. For example, *at_1(1)* has 462 distinct corpus instances for its 112 tasks, with one most common corpus instance occurring in 53 tasks and 27 different corpus instances occurring only once among the 112 tasks. The two averages (position and distance) characterize each corpus instance, but these need to take into account the number of occurrences in further analysis.

7 Future Questions

The discussions in the earlier sections only begin the possibilities where BERTology can be used for lexicography. Below, We describe possible further discussions using BERT MASKing for prepositions (7.1), examining sentences from a different preposition (7.2), retagging training instances (7.3)), and examining differences in the three PDEP corpora (7.4).

- CPA instances that are accurate have nearest CPA instances from groups that are similar and when they are inaccurate do not conglomerate with other instances (7.3)
- Low scores for instances, particularly for CPA instances suggesting an error in the tagging (7.3)
- Can we change a tag because it doesn't fit and will this improve the overall results? (7.3)

- Are OEC and FN query instances nearest to other OEC and FN instances? (7.4)
- Are CPA instances that are accurate are nearest to OEC and FN instances (7.4)
- For instances with the same label: Do the top predictions occur several for the times? Is there possibly a score for these?

7.1 Preposition BERT MASKing

The essence of BERTology is the use of masking a word in a sentence to see similarities with other words. The same is true with single-word prepositions. Moreover, the OEC corpus in PDEP provides an excellent test bed to investigate the examples for each sense.[33] When any corpus sentence is used, five results are generated, identifying different prepositions, with a score between 0 and 1.0. The sum of the five results is 1.0. In some cases, the top score is very close to 1.0; in other cases, there may be several results that are very similar.

Although the MASKing does not identify the senses for each similar preposition, examination of the senses for each possibility will show very close meanings. For example, the spatial sense of *across* (0.40) evokes *over* (0.31), *about* (0.09), *around* (0.07), and *throughout* (0.07). Each of those other prepositions also have spatial senses that can readily be identified, i.e., an appropriately lexicographic task.

Schneider et al. (2022) presents an inventory of adposition supersenses that includes 600 examples in the 52 categories. Each example can be examined using BERTology MASKing. In addition, two efforts have attempted to synchronize the PDEP senses in the supersense field with the guideline categories (Litkowski (2021) and Litkowski (2022)). Each of these studies can benefit from BERTological analysis. In addition, this analysis can also benefit the PDEP substitutable preposition field, initially developed by the lexicographer.

7.2 Sentences from a Different Preposition

The discussion of MASKing (7.1) suggests that we might be able to use a sentence from a different preposition and see if it would be similar to its

[33]Use "*from transformers import pipeline*" and "*unmasker = pipeline('fill-mask', model='bert-base-cased')*" to start the process. Then, use "*unmasker("Briars and thorns tore [MASK] my legs .")* to enter an example sentence, here for *at*.

predictions. We identify similar senses (e.g., spatial senses) and can now examine the corpus instances that have been tagged with those senses. We will see that some of instances will generate the target preposition and some will not. The first constitute a group that are quite similar, while the second suggest significant differences. This analysis may cluster the instances that have been tagged with particular senses and perhaps provide lexicographic justification of the senses.

To examine the similarity to the predictions, we might replace the substitute preposition and then use the nearest neighbor function (e.g., as used in G&S). With the prediction, we can examine where such a substitution would fit with the position and the distance of the top 50 predictions.

7.3 Retagging Instances

The results from the predictions for 8231 test queries also provide the basis for examining the accuracy of the tags for the test queries and the training instances. The test queries are contained in 346 senses for 48 prepositions, 8 of which contain only one sense. For the other senses, we can examine the variations among the several test queries for a single sense and among the different senses for a preposition. In particular, we can use the results themselves to question the accuracy of the tagging in the test query and the several predictions for a query.

In general, we might expect that the scores for the several instances of the same test query would be similar. Instead, a lower than the average score for a single test query might suggest an error in its tagging. For example, the test query for *about_1(1)* (FN instance 369042, "We 'd rowed about it endlessly , but I was quite determined ") might be better tagged as *about_3(2): used to indicate movement within a particular area* rather than *about_1(1): on the subject of; concerning*. This suggestion is made based on having this sense *_3(2)* frequently in the predictions (*labels*), compared with the used of *_1(1)*.

Many of the prepositions have supersenses and subsenses in their sense inventories. Frequently, the predictions of the subsenses have lower coverages and optimums in their scores as compared with the supersense scores. This indicates that it is more difficult to discriminate the nuances in the differences of the subsenses. (here looking at after_1(1)-525)

CPA instances that are accurate have nearest CPA instances from groups that are similar and when they are inaccurate do not conglomerate with other instances

Can we change a tag because it doesn't fit and will this improve the

Test Corpus	**Total**	**FN**	**CPA**	**OEC**
FN Instances	4673	4146 (0.887)	364 (0.078)	163 (0.035)
CPA Instances	2773	667 (0.241)	1844 (0.665)	262 (0.094)
OEC Instances	785	182 (0.232)	164 (0.209)	439 (0.559)
Total	8231	4673 (0.607)	2773 (0.288)	785 (0.105)

Table 7: Correspondence Between Test and First Prediction Corpus

overall results? If a tag is changed in a test query, we expect that (1) more of the training instances will be found with a higher score bringing it to closer than the optimum in ranks dataframe (6.3.1) and (2) an increased coverage and optimums in the scores dataframe (6.3.2). If we modify the tag of a training instance, there are two possible changes: (difficult to say what effect this would have)

7.4 Relations Among the Corpora

Table 7 shows that the first prediction for the test query generally uses a training instance from the same corpus. However, there are some distinguishes in these statistics. The FN instances have a dominant presence from the test queries for each of the corpora, perhaps simply because FN instances constituted the largest instances in the training sets. Similarly, the OCE instances have the smaller presence, again in relation to the smaller instances in the training sets.

Since the quality of the OEC corpus is likely to be the most characteristic instances, but their use in the other two corpora does not have the expected significance, even with those of itself. Would the other OEC instances also be nearest instances?

Are OEC and FN query instances nearest to other OEC and FN instances?

FN test queries would come from a group of sentences that would have the same frame elements. Hence, we would expect that such instances would be high predictions. Is this the case? How would we study this?

Are CPA instances that are accurate are nearest to OEC and FN instances?

8 Possibly Extending BERT to Obtain Structures

The results from such papers as G&S (Gessler and Schneider (2021)) show the value of BERTology. The usual BERTology tasks do not provide additional characterizations of properties such as part of speech and other linguistics properties. We have been searching for other work that might go beyond what has been developing above. The references below might be useful in further research.

Radke et al. focuses disambiguating spatial prepositions, obtaining best results using BERT, enhanced by adding tags to the input text to indicate the presence of place names. They distinguish between geo-spatial, other-spatial, and non-spatial preposition senses. ”We also experiment with a variation on the standard input to BERT in which we tag place name words present in the input text.” ”Extraction of geographic place names and geographic feature types from input sentence to use as features in classifiers” (3.3) ”we experiment here with incorporating an additional feature to indicate that a word in the input text is a named geographic feature” (p. 14) One dataset was derived from the PDEP corpus. (pp. 17ff) Also included other BERT varieties

Espinosa Anke et al. (2021) investigates the possibility of identifying ”lexical collocations in context, categorized into 17 representative semantic categories”, then, performing ”two experiments: (1) unsupervised collocate retrieval using BERT, and (2) supervised collocation classification in context”. The collocations, heavily influenced by Mel’cukian lexical functions, seem to go beyond the usual treatment of BERT characterizations.

Schuster and Hegelich (2022) extract ”coarse features from masked token representations” and predict them ”by probing models with access to only partial information”, to identify variations from ‘BERT’s point of view. In the paper, they ”show how much the differences can be described by syntax but further how they are to a great extent shaped by the most simple positional information.”

Yu et al. (2022) was identified when searching ”Using BERT to construct dictionary definitions”. This study focuses ”on enhancing language model pre-training by leveraging definitions of the rare words in dictionaries (e.g., Wiktionary). To incorporate a rare word definition as a part of input, we fetch its definition from the dictionary and append it to the end of the input text sequence.” This study does not address what was sought, but it does indicate the difficulty of interacting between BERT tasks and lexical information.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL `https://aclanthology.org/N19-1423`.

Luis Espinosa Anke, Joan Codina-Filba, and Leo Wanner. Evaluating language models for the retrieval and categorization of lexical collocations. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1406–1417, Online, April 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-main.120. URL `https://aclanthology.org/2021.eacl-main.120`.

Luke Gessler and Nathan Schneider. BERT has uncommon sense: Similarity ranking for word sense BERTology. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 539–547, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.blackboxnlp-1.43. URL `https://aclanthology.org/2021.blackboxnlp-1.43`.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 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, June 2006. Association for Computational Linguistics. URL `https://aclanthology.org/N06-2015`.

Ken Litkowski. Preposition disambiguation: Still a problem. Technical Report 13-02, CL Research, Damascus, MD 20872 USA, September 2013. URL `http://www.clres.com/online-papers/PrepWSD2013.pdf`.

Ken Litkowski. Pattern Dictionary of English Prepositions. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1274–1283, Baltimore, Maryland, June 2014. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P14-1120.

Ken Litkowski. Supersense V2 Instance Maps. Technical Report 21-02, CL Research, Damascus, MD 20872 USA, 2021. URL http://www.clres.com/online-papers/PSSTMap.pdf.

Ken Litkowski. Tagging the pattern dictionary of english prepositions with preposition supersense examples. Technical Report 22-01, CL Research, Damascus, MD 20872 USA, 2022. URL http://www.clres.com/online-papers/TagPDEP2SST.pdf.

Mansi A. Radke, Abhibha Gupta, Kristin Stock, and Christopher B. Jones. Disambiguating spatial prepositions: The case of geo-spatial sense detection. *Transactions in GIS*, n/a(n/a). doi: https://doi.org/10.1111/2022/tgis.12976. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/tgis.12976.

Nathan Schneider, Jena D. Hwang, Archna Bhatia, Vivek Srikumar, Na-Rae Han, Tim O'Gorman, Sarah R. Moeller, Omri Abend, Adi Shalev, Austin Blodgett, and Jakob Prange. Adposition and case supersenses v2.6: Guidelines for english, 2022. URL http://arxiv.org/abs/1704.02134.

Carolin M. Schuster and Simon Hegelich. From BERT's Point of View: Revealing the Prevailing Contextual Differences. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1120–1138, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-acl.89. URL https://aclanthology.org/2022.findings-acl.89.

Angus Stevenson and Catherine Soanes, editors. *The Oxford Dictionary of English (ODE)*. Clarendon Press, Oxford, 2003.

Wenhao Yu, Chenguang Zhu, Yuwei Fang, Donghan Yu, Shuohang Wang, Yichong Xu, Michael Zeng, and Meng Jiang. Dict-BERT: Enhancing language model pre-training with dictionary. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1907–1918, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-acl.150. URL https://aclanthology.org/2022.findings-acl.150.

A Unused Words

Seventy-eight (78) prepositions were viewed as uncommon, noting 29 (bolded) that seemed to be somewhat common, i.e., having a sufficient number of instances that would support predictions as had been used in the core analysis: ["'cept", "'gainst", "'mongst", 'abaft', '**aboard**', 'absent', 'afore', '**along**', '**alongside**', '**amid**', '**amidst**', 'anent', 'anti', 'apropos', 'aslant', 'astraddle', '**astride**', '**athwart**', '**atop**', 'bar', 'barring', 'betwixt', 'but', '**by**', 'chez', 'come', 'concerning', 'considering', 'counting', 'cum', 'dehors', '**down**', 'ere', 'ex', 'excepting', 'excluding', 'failing', '**following**', 'fornent', 'frae', 'given', 'less', '**like**', 'midst', '**minus**', 'modulo', 'neath', 'nigh', 'notwithstanding', "**o'**", "**o'er**", '**opposite**', '**outside**', 'outta', '**outwith**', 'overtop', 'pace', '**past**', 'pending', '**plus**', 'pro', 'qua', '**re**', 'regarding', 'respecting', '**round**', 'sans', '**save**', "thro'", '**throughout**', 'thru', 'till', '**underneath**', '**unlike**', 'upside', '**versus**', '**vis-a-vis**', '**within**']

B Unused Senses

No needed disambiguation Eight prepositions had only one sense, not needing any prediction, totaling 411 instances: ['besides', 'circa', 'despite', 'except', 'including', 'per', 'since', 'until']

Fewer than 5 instances Fifty-four senses had fewer that 5 instances and were not included in the prediction, totaling 123 instances: ['about_4(3)', 'about_6(n)', 'across_1(1)-1', 'as_4(n)', 'before_4(3)', 'besides_2(n)', 'between_4(4) 5(4a)', 'for_5(4)?', 'from_1(1) 4(3)', 'from_4(3) 12(9)', 'in_10(7a)', 'into_1(1) 2(2)', 'into_9(9)', 'of_18(9)', 'off_7(4)', 'off_8(n)', 'on_1(1) 2(1a)', 'on_21(11)', 'on_23(13)', 'on_6(1e)', 'onto_9(n)', 'over_15(6)-1', 'over_5(2a)', 'over_8(2d)', 'per_2(2)', 'per_4(n)', 'through_12(5) 13(5a) 4(1c)', 'through_3(1b) 10(3)-1', 'through_3(1b) 7(2)', 'to_11(4b)', 'to_12(4c)', 'to_4(1c)', 'to_7(2b)', 'under_13(4e)', 'under_16(5b)', 'under_6(2c)', 'unto_2(2)', 'unto_5(n)', 'unto_6(n)', 'unto_7(n)', 'upon_3(1b)', 'upon_4(1c)', 'upon_6(1e)', 'upon_7(2)', 'upon_9(3a)-1', 'with_14(8a)', 'with_15(9)', 'with_2(2) 10(7a)', 'with_3(2a) 2(2)', 'with_3(2a) 5(3a)', 'with_4(3) 2(2)', 'with_4(3) 5(3a)', 'with_6(4) 9(7)', 'with_7(5) 9(7)']

Few Instances These 26 common, polysemous prepositions have 50 senses that did not occur in prediction files, but did occur in top 50 predictions: ['about_4(3)', 'about_6(n)', 'across_1(1)-1', 'after_10(5a)', 'as_4(n)', 'at_11(6)', 'before_4(3)', 'below_5(n)', 'besides_2(n)', 'from_5(3a)', 'in_10(7a)', 'in_12(9)',

’inside_4(1c)’, ’into_9(9)’, ’of_18(9)’, ’of_8(4)’, ’off_7(4)’, ’off_8(n)’, ’on_21(11)’, ’on_6(1e)’, ’onto_4(n)’, ’onto_9(n)’, ’over_15(6)-1’, ’over_5(2a)’, ’over_8(2d)’, ’per_2(2)’, ’per_4(n)’, ’through_6(1e)’, ’to_11(4b)’, ’to_12(4c)’, ’to_4(1c)’, ’to_7(2b)’, ’toward_5(3)’, ’under_13(4e)’, ’under_16(5b)’, ’under_6(2c)’, ’unto_2(2)’, ’unto_5(n)’, ’unto_6(n)’, ’unto_7(n)’, ’up_4(3)’, ’upon_3(1b)’, ’upon_4(1c)’, ’upon_5(1d)’, ’upon_6(1e)’, ’upon_7(2)’, ’upon_9(3a)-1’, ’with_14(8a)’, ’with_15(9)’, ’with_16(10)’]

Non-PDEP Senses Predictions are generated for 11 senses (labels) that are not in the PDEP inventory. These are, with the 60 number of predictions: ’onto_3(3)’ (1), ’at_1(1) 4(2b)’ (8), ’at_9(5) 11(6)-1’ (3), ’from_4(3) 10(7)’ (4), ’into_1(1) 3(3)’ (2), ’on_8(3) 11(5)’ (4), ’through_1(1) 3(1b)’ (22), ’through_3(1b) 10(3)’ (1), ’through_3(1b) 5(1d)’ (4), ’with_11(7b) 7(5)’ (9), ’with_2(2) 3(2a)’ (2)