FRI Natural language processing course 2022
Project report

University *of Ljubljana*
Faculty *of Computer and Information Science*

Cross-lingual sense disambiguation

Erazem Pušnik, Rok Miklavčič, and Aljaž Šmaljcelj

Abstract

Word sense disambiguation is a problem of identifying the context in which the word is used in a sentence. This can be applied to homonyms, words that sound, or even spell the same way but have completely different meanings. There already exists an equivalent task in English, in this paper attempted to do the same in Slovene language. In this report we describe various methods of context extraction that can be used to determine whether a specific word in a pair of sentences is used in the same context. Through various methods we obtained various results from being worse than the random classifier (*Bag-of-words*) to those who are very similar to a equivalent task in English language (*simplified Lesk* and *BERT*).

Keywords

Natural language proccessing, word context, word sense disambiguation

Advisor: Slavko Žitnik

Introduction

In human language, one word can often be used in multiple ways. It is important to understand word usage to properly develop natural language processing applications. While the problem is easy for humans to solve, since we know from experience of how the world works, computers aren't able to naturally do so.

Word Sense Disambiguation is an important method of dealing with a word context and it involves the use of syntax, semantics and word meanings in context. It can be used in multiple applications, like machine translation, speech processing, information retrieval, text processing and more. The goal of WSD is to ground the meaning of words in certain contexts into concepts as defined in some dictionary or lexical repository. WSD requires two main information sources: context and knowledge. Sense-tagged corpora provide knowledge, leading to data-driven or corpus-based WSD. Use of lexicons or encyclopedias leads to knowledge-driven WSD. Context is established from neighbouring words and the domain of discourse.

While simple approaches like the naive Bayes classifier or Corpus Lesk algorithm, more complex neural network approaches became popular in the 2010s. As an example, Wiriyathammabhum et al. (2012) applied Deep Belief Networks (DBN) and Yuan et al. (2016) proposed a semi-supervised LSTM model with label propagation.

Related work

The English variant of this task is described by Wang et. al. [1] with a Word-in-Context (Wic) task. This task is defined as a binary classification of sentence pairs. Given two sentences who both contain the specified (target) word, the task determines whether the target word is used in the same context (sense) in both sentences.

Dataset used in the mentioned task is also named WiC (ang. *Word-in-Context database*) [2]. It was constructed from example usages in various lexical resources and compiled with constraints of not having more than three instances for some target word and not having repeated contextual sentences across instances. This paper includes a couple of algorithms where results vary. The results are presented with the F-score, which combines precision and recall into a single metric. In this report the score obtained was 71.5%.

In the article [3] the corpus is constructed using stop word removal, which can reduce the size of the corpus up to 25% and improve the accuracy of the model. The article provides insights to two different methods of word sense disambiguation - the shallow approach where the model only considers the surrounding words, and the deep approach where the model is knowledge-based and supervised. The end results of this paper show that the F-score is around 80%.

In the paper [4] the approach to word sense disambiguation is tackled with the use of *Bag-of-words* and

Modified Lesk. The *Bag-of-words* model is used to find out the actual meaning o f a word having different meaning due to different contexts. There is a bag for each sense of an observed word, and all the bags are manually populated. When disambiguation occurs the entire sentence is separated into single words and stop words are removed. After this removal each word would be compared with each word in each bag (where senses are stored). The algorithm searches for the maximum frequency of words in common. The results for the *Bag-of-words* approach in this paper vary based on sample input. On average the F-score is around 66%.

Corpus construction

In the following paragraphs, we will describe our pipeline of corpus construction and context extraction step by step. A general overview of the pipeline can be seen in figure 1.

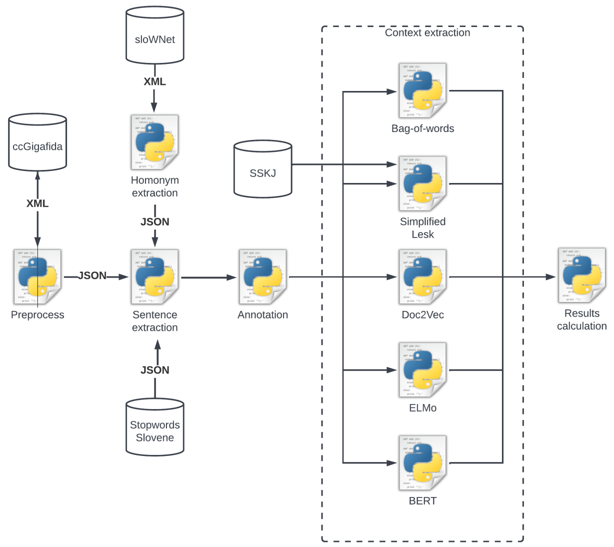
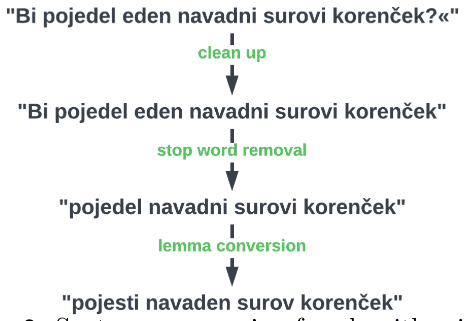


Figure 1. Corpus construction and context extraction pipeline

In order to obtain a list of homonyms, we needed to find a dataset and a way of extraction. A good source of homonyms is *sloWNet* [5], which contains over 70,000 Slovene literals (*Homonym extraction* phase in figure 1). The literal meaning of a word is its original, basic meaning. Each entry always contains at least one English literal and optional Slovenian literals, as well as optional usages and definitions in both languages. Since the file is in *XML TEI format*, we needed to convert it into a *json* file. We decided to group entries by Slovenian literals, where each new entry has its list of Slovenian and English literals, that appeared alongside it, as well as usages and definitions in both languages, in case we will need them in the future. To extract entries, we can sort the list by the number of Slovenian literals each entry has, since one could argue, that the more synonyms each word has, the more likely it is to have multiple meanings. While this is not a perfect way of gathering homonyms, since we still need to handpick them from the top list, it simplifies the process, narrows down the search, and gives us a much better overview of the word's meanings.

We extracted Slovene sentences which contained extracted homonyms from *ccGigafida* corpus [6]. It contains approximately 9% of the *Gigafida* corpus, which is a reference corpus of Slovene language. This 9% proofed enough as for our homonyms, we were able to find at least 1000 pairs of sentences for each specified homonym. In the *preprocessing* step we performed extraction from *XML TEI format* encoding where we took each word of the sentence along with its lemma to construct actual and lemma sentence (*preprocess* phase in figure 1). We also noted the beginning and end index of the observed word as well as its position in the lemma sentence. We stored both pairs of sentences as well as index information in a *json* file.

When we extracted the sentences, we also cleaned up and removed Slovenian stopwords, commonly used words in a selected language with little context information (*sentence extraction* step in figure 1). This way we decrease the computational work required to parse them. As stated from the article [3] the removal of stop words reduced the corpus size up to 25% whereas in our case the percentage removed combined with the simplifying of the text is on average around 52% (depends on the selected word). We obtained the Slovene stop words from the *Github* repository [7]. We converted each word in the sentence to lower case. Then we check if the current word is or contains a punctuation, number or a special character (here we remove words such as abbreviations). Lastly, we check if the word is present in the list of stop words. If any of the above conditions are true, we remove the current word from the sentence. We also noticed it is important to check both the form in which the word is in the sentence and it's lemma form, since some words when converted to its lemma form become stop words (for example: "najboljši" is in the lemma form "dober" which is a stop word). Stop word removal, cleanup of the sentences and lemma conversion, can be seen on figure 2.


"Bi pojedel eden navadni surovi korenček?«"

"Bi pojedel eden navadni surovi korenček"

"pojedel navadni surovi korenček"

"pojesti navaden surov korenček"

Figure 2. Sentence conversion for algorithm input

Alongside the version with removed stopwords, we also kept the original version of the sentences with stopwords as various algorithms required different types of input sentences.

Since we have no automated way of determining if two sentences use the word in same context, we needed to annotate them by hand (*Annotation* phase in figure 1). We marked the pair of sentences as similar, if the target word had the same meaning in both of them. For that we used the definitions in Dictionary of Slovene standard language (SSKJ). In order to ease the annotation process, we wrote a program that provides us with two sentences and accepts the input, whether the target word has same meaning or delete the pair of sentences, in case we determine the context of the word is wrong (e.g. word being used as a name, wrong literal of the word provided, ...).

For context extraction phase (dashed border in figure 1) we tried several methods in order to compare different methods by their accuracy. The same was done in our base article where authors compared the accuracies of different context extraction methods and embeddings. The result of all implemented methods is an updated initial corpus with information on whether the target word in a pair of sentences is used in the same context.

As a basic and simple method, we implemented *Bag-of-words* method in order to get a baseline on which we improved our result. For this method we used sentences with already removed stopwords. We stored words that appeared around the target word and represented them as one big vector. We counted the number of occurrences of these words in order to construct a vector for each sentence. In order to determine the use of a target word in a pair of sentences as the same, we calculated the cosine similarity between both vectors and applied the thresholding method. So, our *Bag-of-words* method accepts the following parameters:

- Window size: defines how many neighbor words around target word we will use as context.
- Cosine distance threshold: if cosine similarity of a pair of vectors is bigger than this predetermined threshold, we mark the pair as having same context.

The second method we tried is the *simplified Lesk* algorithm. We used the corpus of sentences with their stopwords removed. In order to implement the algorithm, we obtained the entire Dictionary of Standard Slovene Language (SSKJ), accessible in a Github repository[1]. We compared lemma forms of words that appeared in sentences with glosses for different meanings. In the preprocessing step of obtained glosses we grouped them by the target word. We also transformed words in glosses into their lemma form in order to compare them with our pairs of sentences. For that, we used *CLASSLA* pipeline [8] to get lemma forms of words in glosses and stored them in a *json* file.

[1]https://github.com/van123helsing/SSKJ

Next context extraction method we tried is *Doc2Vec* which is an extension of *Word2Vec* embedding. We tagged every word from all the extracted sentences and trained a *Doc2Vec* model. We used sentences that contain stopwords. Then we looped through pairs of sentences and gave the first sentence of the pair as a new sentence to the model, which it then computed similarity to other sentences already present in the model. We extracted the computed similarity to the other sentence in the pair and if the computed similarity was above a predefined threshold, the contexts were deemed similar. The *Doc2Vec* method accepts the following parameters:

- Vector size: dimensionality of the feature vectors.
- Windows size: defines how many neighbor words around target word we will use as context.
- Min count: defines the minimum frequency of a word for it to not be ignored.
- Epochs: Number of iterations (epochs) over the corpus.

Figure 3. *ELMo* (on the right) and *BERT* (on the left)

We also used *ELMo* method for context extraction, which is shown in picture 3. *ELMo* is a deep contextualized word representation that models both complex characteristics of word use (e.g., syntax and semantics), and how these uses vary across linguistic contexts (i.e., to model polysemy). These word vectors are learned functions of the internal states of a deep bidirectional language model (biLM), which is pre-trained on a large text corpus. They can be easily added to existing models and significantly improve the state of the art across a broad range of challenging NLP problems, including question answering, textual entailment and sentiment analysis [9]. Before using the method, we had to define the following parameters:

- Euclidean distance threshold: if Euclidean distance of a pair of vectors is smaller than this predetermined threshold, we mark the pair as having the same context.

The last method we used was *BERT* (pictured on figure 3), which stands for Bidirectional Encoder Representations from Transformers. For each homonym, we generated a context-based embedding. Embeddings are simply (moderately) low dimensional representations of a point in a higher dimensional vector space. In the same manner, word embeddings are dense vector representations of words in lower dimensional space. Once we had the embeddings, we looped through the pair of sentences and calculated the cosine distance between their embeddings. In order to determine, if the sentence uses our homonym in the same context, we compared the distance to predetermined threshold. If the computed similarity was above the predefined threshold, the contexts were deemed similar. When running the *BERT* method, we defined the following parameters:

- Cosine distance threshold: if cosine similarity of a pair of vectors is bigger than this predetermined threshold, we mark the pair as having the same context.
- Model: defines which *BERT* model our program will use. Either a local path or just the model name can be provided. If we provide a model name, the script will try to download the model.

Results

We needed to pick homonyms, in such a way, that would allow us to easily distinguish between meanings in a sentence, since we needed to annotate each pair by hand. We also chose words which have a lot of different meanings recorded in the Dictionary of Slovene standard language. This way we picked the following homonyms for testing our methods: *list*, *postaviti*, *surov*, *prst*, *klop*, *tip* and *tema*.

For example, the word *postaviti* has eight different meanings in different contexts, *tip* has four, *list* has six, and so on. Not all of different meanings of selected words are present in our corpus, but interestingly the word *tip* has all its meanings present in our corpus. For each homonym we annotated around 100 cases, which means we compared whether the context usage of a word across both sentences was the same in 200 sentences. This resulted in around 800 cases for our data set and therefore around 1600 sentences. The best results for each method are shown in table 1.

Best results for *Bag-of-words* were obtained on windows size 2 and cosine similarity threshold of 0.6. Results for *Bag-of-words* produced expected evaluation results. It only labeled a pair of sentences as the same context when words around the target word were the same. This is successful only in identifying figure of speech (eg. *s prstom pokazati*) but is unsuccessful in other instances, hence low recall. This method was mainly used as a baseline in order to show that *Bag-of-words* is worse than the random classifier and that other methods are significantly better.

Method	F_1	**Recall**	**Prec.**	**Acc.**
Bag-of-words	0.041	0.021	0.786	0.334
Simplified Lesk	0.721	0.665	0.788	0.652
Doc2Vec	0.75	0.838	0.679	0.622
ELMo	0.780	0.79	0.769	0.698
BERT	0.845	0.722	0.781	0.677

Table 1. Results of different context extraction methods

For *simplified Lesk* algorithm, we didn't pass any input parameters as it simply assigns a context that most words from a specific gloss appear in. The only problem is that if a sentence didn't contain any words from glosses, the algorithm assigned the context of the first gloss as it is the most common one. This may lead to a pair of sentences which both don't contain any words from glosses, and an algorithm assigns the first context to both of them, which may not be true. The target word could indeed be used in the same context in a pair, but not necessarily the first one, hence the classification might be correct, but only by chance.

With *Doc2Vec* best results were obtained with threshold value at 0.3. For model parameters, we used *vector_size* value of 100, *window* 2, *min_count* 1 and *epochs* value of 100. We can see that we produced much better recall but worse precision and accuracy, but the overall F_1 score is better than in *simplified Lesk*.

ELMo results were obtained using the threshold value 1.7. It is important to note we used the Euclidean distance, therefore we aim for a lower threshold rather than a bigger one. The results are worse when compared to *BERT*. This is expected as the *BERT* was trained on a Slovene corpus and *ELMo* was not. *ELMo* performs better than Simplified *Lesk* and better than *Doc2Vec* even without Slovene training data set. The Accuracy of *Elmo* is the highest of all methods, but the F_1 score is worse than that of *BERT* which indicates that *BERT* performs slightly better. In the figure below 4 we see the clustering of the examples based on colours. Every pair of colours was correctly classified except for the green colour. The yellow colour had the Euclidean distance higher than the threshold and was therefore classified as False.

Lastly, we tested the *BERT* method with *CroSloEngual BERT* model [10]. We wanted to use *SloBERTa* model [11], but since the repository does not include the *vocab.txt* file, which contains all the vocabulary words, we could not get it to run with our script, which depends on it. The *BERT* method clearly produced the best results. While we get worse recall than with *Doc2Vec*, we make up for it with the best F_1 score. The problem

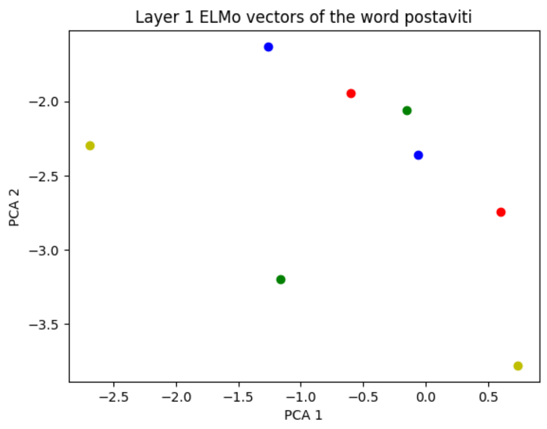


Figure 4. Visualisation of *ELMo* results for examples of the word *postaviti*

with this method is probably the fact, that it relies on its vocabulary. Since the model wasn't trained on just Slovenian language, we noticed, that it is missing some of the common words. We tested different values of threshold, but the best performing one was 0.4.

We also wished to determine, which of the homonyms tested, was the worst performing overall. For each homonym we ran all of the methods and found out, that the homonym *postaviti* is the hardest to guess correctly. The results can be seen in table 2. This could be due to the fact, that the homonym *postaviti* has eight different meanings, as we had mentioned before.

postaviti				
Method	F_1	**Recall**	**Prec.**	**Acc.**
Bag-of-words	0.0	0.0	0.0	0.609
Simplified Lesk	0.349	0.357	0.353	0.5
Doc2Vec	1.0	0.398	0.57	0.41
ELMo	0.667	0.93	0.519	0.636
BERT	0.837	0.429	0.567	0.5

Table 2. Results on different context extraction methods on homonym *postaviti*

We also compared results that we obtained with our various methods with the results given in the base article [1]. Results from this article are shown in table 3 in accuracy metric. We can see that we were able to produce slightly worse results. All our methods performed worse than base articles' best method, but we were in reasonable range between their implementation of *Continuous Bag of Words* and *BERT*. Even though a straight comparison of the evaluation metrics is possible, as the authors of the English variant of this task used a completely different corpus to obtain those results.

Model metrics	**WiC Acc.**
Most Frequent Class	50.0
CBOW	55.3
BERT	74.9
BERT++	74.9

Table 3. WiC task performance in base article

Conclusions

Throughout this project we tried several methods to extract context from a pair of sentences and determine whether a specific word is used in the same context. Our implemented methods produced satisfying results with our best method performing much better than a random classifier.

During our work we noticed many areas for further research and development. One option would be to continue development of *BERT* model, by splitting our corpus into training and testing data, and using it to further train the model. This would probably result in overall better results, but would require much more hand-annotated data.

However, the obtained corpus of all described methods can't already be used for further processing tasks as the accuracy is not high enough, so some hand annotation and checks are needed.

References

[1] Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32, 2019.

[2] Mohammad Taher Pilehvar and José Camacho-Collados. Wic: 10, 000 example pairs for evaluating context-sensitive representations. *CoRR*, abs/1808.09121, 2018.

[3] Madeeh Nayer El-Gedawy. Using fuzzifiers to solve word sense ambiguation in arabic language. *International Journal of Computer Applications*, 2013.

[4] Alok Pal, Anirban Kundu, Abhay Singh, Raj Shekhar, and Kunal Sinha. An approach to word sense disambiguation combining modified lesk and bag-of-words. volume 3, 02 2013.

[5] Darja Fišer. Semantic lexicon of slovene 3.1, 2015. Slovenian language resource repository CLARIN.SI.

[6] Nataša Logar, Tomaž Erjavec, Simon Krek, Miha Grčar, and Peter Holozan. Written corpus ccGigafida 1.0, 2013. Slovenian language resource repository CLARIN.SI.

[7] Gene Diaz. Stopwords slovene (sl), 2016. Collection of stopwords for the slovene language.

[8] Nikola Ljubešić and Kaja Dobrovoljc. What does neural bring? analysing improvements in morphosyntactic annotation and lemmatisation of Slovenian, Croatian and Serbian. In *Proceedings of the 7th Workshop on Balto-Slavic Natural Language Processing*, pages 29–34, Florence, Italy, August 2019. Association for Computational Linguistics.

[9] AllenNLP - ELMo — Allen Institute for AI — allenai.org. https://allenai.org/allennlp/software/elmo. [Accessed 25-May-2022].

[10] M. Ulčar and M. Robnik-Šikonja. FinEst BERT and CroSloEngual BERT: less is more in multilingual models. In P Sojka, I Kopeček, K Pala, and A Horák, editors, *Text, Speech, and Dialogue TSD 2020*, volume 12284 of *Lecture Notes in Computer Science*. Springer, 2020.

[11] Matej Ulčar and Marko Robnik-Šikonja. Slovenian RoBERTa contextual embeddings model: SloBERTa 2.0, 2021. Slovenian language resource repository CLARIN.SI.