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

David Miškić, Kim Ana Badovinac, Sabina Matjašič

Abstract

Keywords
sense, disambiguation

Advisors: Slavko Žitnik

Introduction

Many words have multiple meanings, which are determined by their context. We can determine the specific meaning with the help of word-sense disambiguation (WSD). The problem has been long active in the field of natural language processing, as those languages are context-dependent and understanding is not possible without knowing the meaning [1]. The hardness of the problem has been described as equivalent to solving central AI problems [2]. WSD relies on knowledge, but those can vary considerably depending on corpora, and there are different approaches with their own standardization. But the need for automatic knowledge is growing with the amount of unstructured data and the use of machine translation. There are multiple approaches we will explore in the following section, some of those aim to disambiguate a text in one language by exploiting its differences with another language in a parallel corpus. This is cross-lingual word sense disambiguation (CLWSD).

Related Work

Manual encoding

Wilks, 1972

Preference semantics assumed that each sense has a formula that could determine the meaning. The formula was composed of hierarchical selectional constraints, but those could be loosened within some conditions. In addition, verbs and adjectives listed the preferred sense of the words they modified. Disambiguation involved choosing a formula that was satisfied the most.
One example is the use of the verb to drink in sentences ”The adder drank from the pool” and ”My car drinks gasoline”. The verb prefers animate subject. In the first case, the adder can have the meaning of machine or animate object, so the animate meaning satisfies the verb’s condition. In the second case, the car is not animate so it does not fit the verb. The meaning of the verb has to be extended and a new usage is added. In this way, the meanings can be dynamically evolved over the text [1].

Small 1980

Word expert parsing presumes the lexicon to be the source of information, but the representation is build up from disambiguation as well. Small’s unconfirmed hypothesis was that human knowledge is more word-based rather than rule-based, so each such word would need its own disambiguator [1].

Dictionary based

Lesk 1986

Optimizing dictionary definition overlap with the help of the dictionary, where the sense can be determined depending on the sense of the words close to it, based on dictionary definitions. One example is the word ”pine cone” where pine is either a tree or a verb, while the cone is either a body or a fruit. The overlap is in the words evergreen and a tree in the dictionary definition, so those meaning were picked (pine as tree and cone as a fruit) [1].

Connectionist based

Waltz and Pollack 1985

The approach was to carry out syntactic and semantic processing in parallel, based on psycholinguistics observation. Such an approach attempted to model the errors as well, one example is the sentence ”The astronomer married the star” where humans make a temporary mistake of regarding the word star as a celestial body before the more probable sense of celebrity is picked because the semantic effect of the word astronomer is stronger than case selection of the verb to marry. The model is composed of a network of nodes, representing case frames, semantic priming, and syntactic preferences [1].

Cross Lingual

Sense projection (SP) approach can bootstrap the creation of sense-annotated parallel corpora by exploiting existing

resources in well-represented languages, with word alignment and connected sense inventories as the only requirements. [3]

Unlike SP, multilingual sense intersection (SI) approach does not require any of the texts in a parallel corpus to be sense-annotated, so it can be applied to a wider range of existing resources. Its logical foundation is in that a polysemous word in a language is likely to be translated into different words in other languages, so the comparison with the semantic space of each translation help select the sense actually intended. [3]

Dataset description

Selection of starting candidate ambigous words was made by using a Slovene dictionary of Homonyms [4]. We collected the corpus from the written corpus ccGigafida [5]. This corpus contains only approximately 9% of the Gigafida corpus, but still contains many various themed documents. Using a list of known ambigous words, we collected the sentences including these words from the ccGigafida.

Subsequently we wanted to get suggestions for more candidates for ambigous words based on the text we have. From existing ccGigafida corpus or any other text source that we have chosen, we check every plausible word for ambiguity. We determine if the word is ambigous by checking its input in the web dictionaries SSKJ [6] or PONS [7]. If the word has multiple entries in the dictionary and its descriptions are uniquely different, that word is suggested as ambigious.

Methods

Preprocessing data

After all the corpora were gathered we had to process it so that everything had an uniform form. Tokenization is a common task in NLP. It is a way of separating a piece of text into smaller units called tokens. Here, tokens can be either words, characters, or subwords [8]. The token occurrences in a document can be used directly as a vector representing that document. We used a Slovene tokenizer from classla library in Python and we also removed Slovene stopwords which we downloaded from nltk Python library.

TF-IDF

For feature extraction, we used the TF-IDF method, which is short for term frequency-inverse document frequency. It is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. It is often used as a weighting factor in searches for information retrieval, text mining, and user modeling. The TF-IDF value increases proportionally to the number of times a word appears in the document and is offset by the number of documents in the corpus that contain the word, which helps to adjust for the fact that some words appear more frequently in general [9]. We used vectorization, where each tokenized sentence in our corpus is vectorized with unigrams and bigrams. The query is transformed into a vector and then compared with vectors of the corpus. To measure similarity between compared vectors we computed cosine distance, and the most similar is chosen.

word2vec

Word2vec is a technique for natural language processing. It is a type of word embedding which is one of the most commonly used representations of document vocabulary. Word embeddings represent individual words as real-valued vectors in lower-dimensional space [10]. The word2vec algorithm uses a neural network model to learn word associations from a large corpus of text. Once trained, such a model can detect synonymous words or suggest additional words for a partial sentence [11]. In our case, we compared input sentences with sentences in data collection. Then we break sentences down into words. For words in each sentence, we calculate smooth inverse frequency and SIF with similarity being the sum of SIF* vector for each word and its vector. This is how we achieved that all word embeddings can be combined into one. Finally, we compared these sentence embeddings with cosine similarity. If the new word is not in the model, we used translation to English with google translate (so if a word has more senses, it might translate to one) and take the word in the model that is most sense similar based on the Leacock Chodorow method [12].

Neural network

We trained a neural network utilizing Long Short-Term Memory (LSTM) layers [13]. LSTM networks are a type of recurrent neural network capable of learning order dependence in sequence prediction problems. LSTMs are explicitly designed to avoid the long-term dependency problem, which is achieved with a chain structure that contains four neural networks and different memory blocks called cells [14]. To train our model, we used pre-trained embeddings and precomputed word frequencies [15]. We increased the amount of vocabulary by generating similar tokenized sentences with word2vec.

References

[1] *The Oxford handbook of computational linguistics*. Oxford University Press, 2004. Bibliografija ob posameznih poglavjih Kazali.

[2] Roberto Navigli. Word sense disambiguation: A survey. *ACM Comput. Surv.*, 41(2), feb 2009.

[3] Francis Bond and Giulia Bonansinga. *Exploring Cross-Lingual Sense Mapping in a Multilingual Parallel Corpus*, pages 56–61. 01 2015.

[4] Júlia Bálint. *Slovar slovenskih homonimov: na podlagi gesel Slovarja slovenskega knjižnega jezika*. Znanstveni Institut Filozofske Fakultete, 1997.

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

[6] Fran/iskanje, April 2022. [Online; accessed 29. Apr. 2022].

[7] Pons slovar, April 2022. [Online; accessed 29. Apr. 2022].

[8] What is Tokenization | Tokenization In NLP, July 2021. [Online; accessed 29. Apr. 2022].

[9] Anand Rajaraman, Jure Leskovec, and Jeffrey D. Ullman. Mining of Massive Datasets. *Mining of Massive Datasets*, January 2014.

[10] A Gentle Introduction to the Bag-of-Words Model, August 2019. [Online; accessed 29. Apr. 2022].

[11] A Beginner's Guide to Word2Vec and Neural Word Embeddings, October 2021. [Online; accessed 29. Apr. 2022].

[12] Claudia Leacock and Martin Chodorow. Combining Local Context and WordNet Similarity for Word Sense Identification. *WordNet: An Electronic Lexical Database*, 49(2):265—283, January 1998.

[13] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-term Memory. *Neural Comput.*, 9(8):1735–80, December 1997.

[14] Understanding LSTM Networks – colah's blog, January 2022. [Online; accessed 29. Apr. 2022].

[15] jvparidon. subs2vec, April 2022. [Online; accessed 29. Apr. 2022].