Homework 3: WSD of Word-in-Context Data

Simone Rossetti
rossetti.1900592@studenti.uniroma1.it

1 Introduction

In this report we are going to present a possible solution to accomplish simultaneously two very common tasks in NLP, Word-in-Context Disambiguation (WiC) and Word Sense Disambiguation (WSD). Given a polysemous lemma we want to discriminate weather it has the same meaning in two different sentences, and discriminate among the all possible senses for that lemmata in context.

2 Dataset and Pre-processing

The dataset chosen (Raganato et al., 2017) is a unified five standard all-words Word Sense Disambiguation datasets. This is an evaluation framework in English and all the sense annotations belong to WordNet 3.0 sense inventory. In particular the training corpora used in this specific case is the unified version of SemCor and OMSTI training sets, which amount to a total of more than 1M instances.

The training corpora is a gold standard, each sentence and instance contains all the relevant information needed, such as lemmatized words and Universal POS (Part-of-Speech) tagging. Furthermore the sentences are well filtered and tokenized and do not require a particular pre-processing step.

To deal with the sense annotated data I relied on the NLTK library APIs dedicated for WordNet 3.0, which is a lexical database made of lemmata and synsets hierarchical trees.

3 Vocabulary and Context Embedding

I decided to use a pre-trained context embedding model based on attention, namely one of the most largely used, in particular the feature extraction side of Bidirectional Encoder Representations from Transformer (BERT) (Devlin et al., 2019). There are many advantages in using context embeddings from BERT, first of all as any other transformer it is based on attention (Vaswani et al., 2017), which has been shown to be a breaking through in NLP and many other fileds during the years, but also the fact that BERT uses WordPiece (Wu et al., 2016), which permits to encode all possible words with a relatively small vocabulary made of about 30K words between most frequent words and word-pieces. This means in practice we do not need to use a particular fixed size vocabulary and we do not have to deal with Out-of-Vocabulary (OOV) words.

The BERT Model and Tokenizer (a module which performs tokenization according to the word piece vocabulary) are available in many open-source libraries, the one I choose and which is very well documented is the Python library `transformers`[1]. In particular I adopted the pre-trained `bert-large-uncased` variant.

4 WSD Methodology

To address the supervised Word Sense Disambiguation task I implemented and tested two main approaches based on Lexical Knowledge Bases (LKBs). Note that no transformer architecture has been fine-tuned due to computational power and time limits, furthermore only transformers encoder part have been used.

4.1 Gloss-based Method

The first solution is based on context comparison between the contextualized polysemous lemma and the relative glosses definitions embedded into vectors via BERT encoding and BiLSTM features extraction and performing multi-class classification as in (Kumar et al., 2019), but in a simpler fashion, as a compromise with (Huang et al., 2020), which instead perform binary classification discriminating between sentence and gloss definition encoded and pooled by BERT encoder.

[1] https://huggingface.co/

The architecture takes as input a batch of samples made by a pair of sentence and the query word synset relative glosses, before using the BERT Tokenizer the lemma is added at the beginning of each gloss, as in (Huang et al., 2020), while the query word in the sentence is surrounded by some special characters in order to retrieve the splitted word indices. At this point the algorithm proceeds similarly to what is described by these authors (Du et al., 2019), but with the simple contribution of using a BiLSTM to encode the gloss sense definition instead of training the full BERT model. Once the sentence and the glosses are fed into the BERT encoder, the contextualized embeddings of the word to be disambiguate are taken (if the word is splitted then an average or sum is performed). Glosses are instead treated differently, once the full sequence (of a prefixed maximum length) for each gloss is encoded via BERT as stated before I used a BiLSTM to produce an embedding of the gloss. At this point an MLP layer processes the embeddings which are stacked and finally, after activation are dot multiplied by the contextualized word to get the logits distribution over the sense glosses. Finally a softmax activation is used to transform it into probabilities. This architecture reached poor results compared with respect to the following one as shown in the Figures 1 and 2. Please notice also that equations and schemes are not reported since the cited papers are quite exhaustive about that and there is no needing in being repetitive.

4.2 Graph-based Method

The second one relies on a intuitive and faster-to-train solution which fully exploit the BERT contextualizing capability to discriminate a word sense over the WordNet synsets vocabulary (Bevilacqua and Navigli, 2020); once the word is encoded within its context and its last 4 hidden states are extracted and summed or averaged from the BERT encoder (if there are multiple word piece an average or sum is performed between those), we discriminate its sense by the use of Multi-Layer-Perceptrons (MLPs) and the use of an Adjacency Matrix built upon the graph of the WordNet synsets hyponyms and hypernyms, avoiding the closures as suggested in the same reference paper, to learn the BERT context encoding and condition the classification by adding probability contribution of adjacent classes, weighted by the relative number of connections of each (the more edges a synset has the less relevant is). During inference only synsets relative to the polysemous word are considered, unrelated classes probabilities are zeroed. The output layer is randomly initialized, while instead in (Bevilacqua and Navigli, 2020) and (Kumar et al., 2019) it is shown how much beneficial is initializing it with a embedded representation of each synset and its glosses. It results in a fast-to-train and acceptable WSD classifier, indeed in one hour of training it covered and over-fitted the whole SemCor+OMSTI training set (3, achieving 76.8% accuracy on test set, Figure 4 and 76.7% F1 score, Figure 5.

4.3 WiC Methodology

To train a model based of transformer for WiC task requires the use of a large training set. Indeed I created a custom dataset which map SemCor+OMSTI WSD training set into a training set following the format style of WiC. Since the Graph-based Method resulted in being the more effective and faster than the other, I built upon it a Word-in-Context head which takes the swish activated output from the WSD with Graph model described above. This layer indeed encodes the the learnt features of the words in context learned by the first linear layer of the WSD module. The WSD module is freezed.

The WSD extracted features are then treated separately by a MLP layer, with common weights, after batch normalization and swish activation they are concatenated and fed to a final MLP classifier which infer similarity by the use of a binary cross entropy loss. The network WiC training are reported in Figure 6 and 7, reaching 67.8% accuracy.

5 Results

Trainings are stopped once overfitting is reached and best models are saved during epochs.

Model	Acc	F1
BERT+Gloss	.661	.670
BERT+Graph	**.768**	**.767**

Table 1: Main results on the two approaches.

Params	Value
Architecture	BERT+Graph
Hidden sizes	512
Dropout	0.3
Optimizer	Adam
Learning rate	5e-4
Batch size	200
Classes	117659

Table 2: Final model parameters.

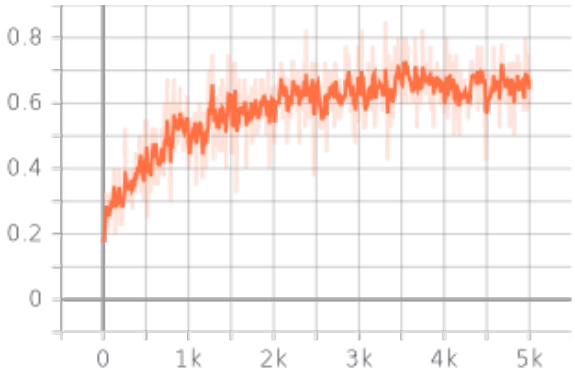

Figure 1: WSD Gloss-based Method training accuracy.

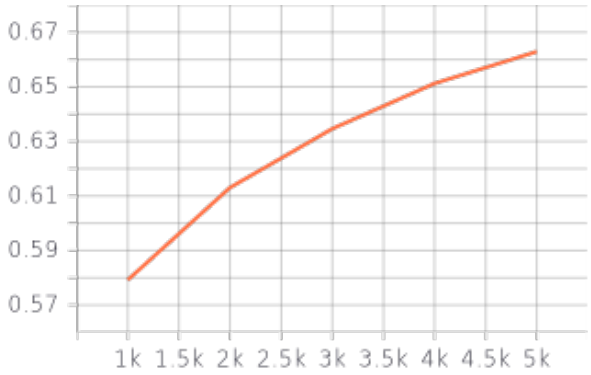

Figure 2: WSD Gloss-based Method validation accuracy.

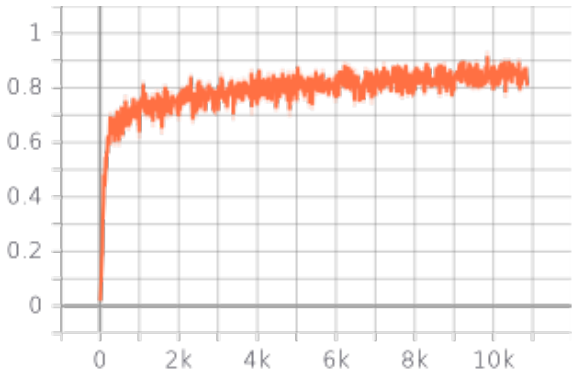

Figure 3: WSD Graph-based Method training accuracy.

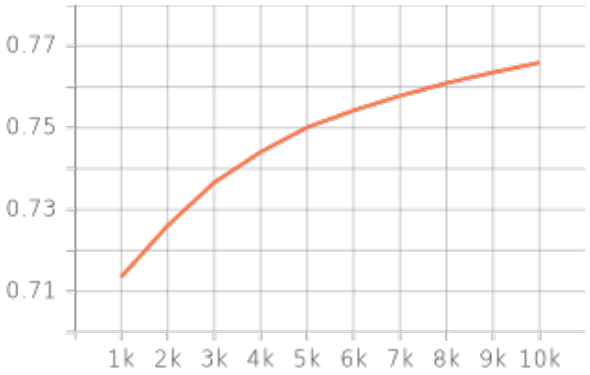

Figure 4: WSD Graph-based Method validation accuracy.

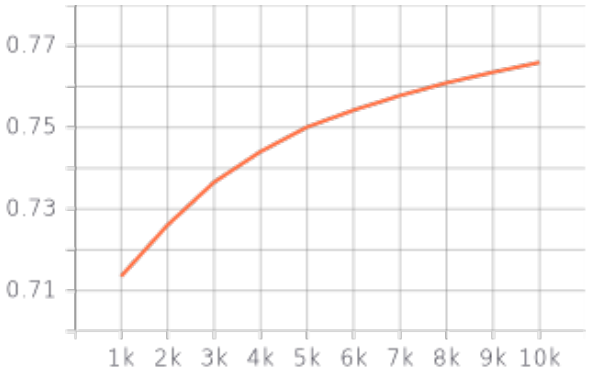

Figure 5: WSD Graph-based Method validation F1 score.

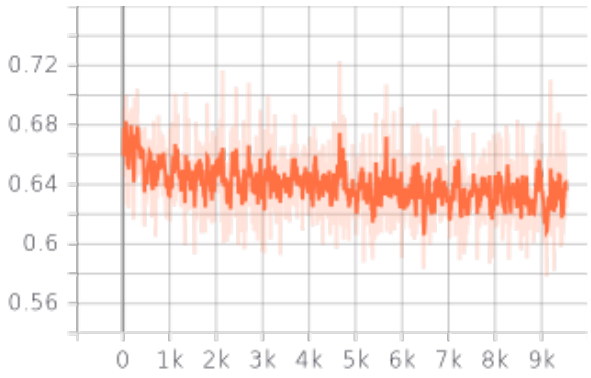

Figure 6: WiC + WSD Graph-based Method, WiC training loss.

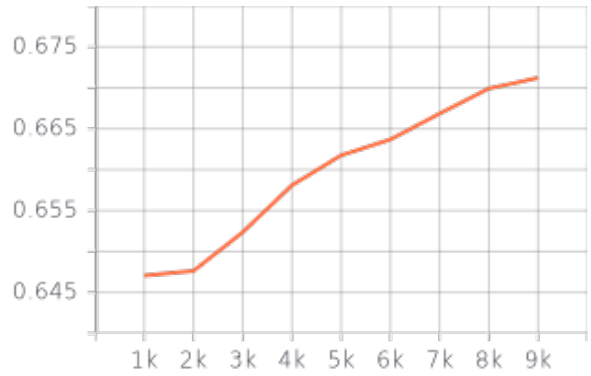

Figure 7: WiC + WSD Graph-based Method, WiC validation accuracy.

References

Michele Bevilacqua and Roberto Navigli. 2020. Breaking through the 80% glass ceiling: Raising the state of the art in word sense disambiguation by incorporating knowledge graph information. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2854–2864, 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.

Jiaju Du, Fanchao Qi, and Maosong Sun. 2019. Using bert for word sense disambiguation.

Luyao Huang, Chi Sun, Xipeng Qiu, and Xuanjing Huang. 2020. Glossbert: Bert for word sense disambiguation with gloss knowledge.

Sawan Kumar, Sharmistha Jat, Karan Saxena, and Partha Talukdar. 2019. Zero-shot word sense disambiguation using sense definition embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5670–5681, Florence, Italy. Association for Computational Linguistics.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation.