Word-in-Context Disambiguation

Pavlo Vasylenko
vasylenko.1880258@studenti.uniroma1.it

1 Introduction

Word-in-Context (WiC) is a challenging task where we need to define whether a given lemma means the same in two separate sentences. The problem can be tackled by use of Bi-LSTM which encode contextual information into output vector of a given word. Pipeline of the proposed solution consists of training two seperate models: contextual encoder (Bi-LSTM) and WiC classifier (multi-layer perceptron (MLP)). Such separation helps to get more useful contextual embeddings which are, then, used for classification.

2 Model

Our model employs pre-trained word embeddings and consists of two main parts: contextual encoder (CE) producing contextual embedding for a given word in each sentence and WiC classifier Fig. 1.

2.1 Word embeddings

We have chosen GloVe (Pennington et al., 2014) with vocabulary size of 400000 which makes out of vocabulary (OOV) words constitute only 0.77% of all words in the corpus Fig. 3. OOV words are replaced by special token $< unk >$ which embedding, instead of initialized randomly, is an average of first $N = 50000$ vectors of vocabulary.

2.2 Contextual Encoder

The original dataset is small in terms of number of words for which we use Bi-LSTM output vector to be considered as contextual encoding, that is, we use only one word per one sentence. If we train recurrent network on original representation of dataset, it might be not enough to learn how to represent a word in context. To overcome this problem, we can first train a model that encodes contextual information and, then, use its output embeddings for later comparison. For this purpose, a good option is to use a language model (Yuan et al., 2016) which is trained to predict held-out word masked by special token.

Inspired by negative sampling (Mikolov et al., 2013), we modified training procedure by replacing K-classification of a hidden word to binary classification where we concatenate LSTM output with either an embedding of a real target token (positive sample), corresponding to label 1, or with a randomly sampled word (negative sample), with label 0. In nutshell, the model predicts if a word fits to context. Such a choice gives advantage in training since we reduce complexity of model having only two classes instead of vocabulary size. In addition, the presence of random negative samples in training mode makes better generalisation of the model.

We also tried approach when instead of concatenation of two vectors, they are compared by cosine similarity and the result is mapped to [0, 1] interval by transformation: $\frac{1+cosine(v1,v2)}{2}$. Such approach, if successful, benefits from reducing model, since then we just could measure cosine between two contextual vectors, thus no need in WiC classifier step Fig. 2.

2.3 WiC classifier

The classifier simply takes two embeddings produced by contextual encoder, concatenate them and feed to MLP with two hidden layers. Since we can incorporate information about part of speech (POS), we tried two different approaches: adding POS embeddings to vector to be classified and multi task learning with predicting POS tags (Raganato et al., 2017). The later one approach takes output of first hidden layer and feed it to POS classifier, hence first hidden layer learns to perform two tasks encode information for the main classification and POS classification Fig. 4.

3 Data

For training Contextual Encoder, we construct additional dataset that is composed of one sentence where we replace randomly any word with special token $< masked >$. With probability 0.5 we choose whether we use a real token or we sample a token from the dataset corpus. Such negative sampling is done so a probability of occurrence of a word is the same as for occurrence it in training data. It is worth to mention, that by using one sentence per time we double the size of dataset to become 16000 samples which helps in training.

3.1 Preprocessing

Stop words usually do not contain much information, thus we can remove them to focus on more important information (Khanna, 2010). It also reduces complexity since we shorten sentences.

3.2 Data augmentation

In order to improve generalization of tested models, we tried two different methods to enrich the dataset: random swap of words (ES, 2021) and random replacement of words. In both cases, the maximum number of changed words is set in percentage of sentence length, then, based on that threshold, a random number of swaps or replacements is chosen. In such way we also have different number of changes even for the same sentence length. It worth to mention that we replace words by sampling from vocabulary that we faced in training dataset, the same procedure as we described above for negative sampling.

4 Expiriments

4.1 Baseline

We use MLP with two hidden layers as a baseline (see Table 1). Input word embeddings are averaged separately for each sentence and obtained two vectors are concatenated to be fed to MLP. To overcome overfitting, along with dropout, we use random replacement of tokens with maximum of 20% which is also slightly reduces wiggling of validation loss and accuracy metric Fig. 5.

4.2 Context Encoder

The model is build of two staked Bi-LSTM and two hidden-layer MLP with batch normalization and dropout Table 2. MLP can be trained in two modes as a classifier or as a layer that transform vector to a different dimension so it can be compared, via cosine similiarity, with input embeddings. To reduce overfitting, we swap words with threshold of 30% and random replacement of 10%. Words swap is more beneficial for that task since it does not change meaning so much.

4.3 Word in context classifier

We trained WiC classifier for three different configuration related to POS tags: without POS information, concatentation POS embeddings (CE + MLP + POS), multitask learning (CE + MLP + Multi-learning). For first two approaches, classifier consists of two hidden layer MLP with batch normalization and ReLu activation Table 3. Multi task learning additionally contains POS classifier layer producing extra cross entropy loss to be added to main one.

4.4 Model with extra POS tags from NLTK library

In addition, we tried to improve performance by adding extra information of POS tags for all words in sentences. Before fed to Bi-LSTM, trainable POS embeddings are concatenated to word embeddings. However, for this task, we do not remove stopwords because as we think information about POS, in that case, can be beneficial, since some stopwords might belong to different POS.

5 Results

Results for Contextual Encoder and original WiC problem are shown in Table 4 and 5.

6 Conclusion

We showed that Bi-LSTM can perform better than MLP with simple averaging of word embeddings, however it might require a pre-trained language model to encode the context. Pipeline of CE + Cosine showed that the model pure for learning that representation that can be measured with cosine similarity. No success achieved with multi-learning in WiC classifier. It might be better to use multi-learning in CE to have better context embeddings.

Adding extra data in form of POS tags do not boost performance, which might be that CE learnt to make predictions mostly on POS tags but not on overall context which is disadvantageous for WiC classifier. Moreover, it makes sense to try multi-learning with POS tags in CE part rather than how we did in WiC classifier.

Setting	Parameter
MLP layers	2
MLP BatchNorm	present
MLP Activation	ReLU
MLP dropout	0.6
Embed. dimension	50
Replaced words treshold	20%
Stopwords removal	Yes

Table 1: Baseline configuration

Setting	Parameter
Word embeddings	GloVe 300
LSTM layers	2
Staked LSTMs	2
Staked LSTMs	2
LSTM dropout	0.7
MLP layers	2
MLP BatchNorm	present
MLP Activation	ReLU
MLP dropout	0.7
Swapped words treshold	30%
Replaced words treshold	10%
Stopwords removal	Yes
Optimizer	Adam lr=0.0001

Table 2: Context Encoder configuration

Tables and Figures

Setting	Parameter
Word embeddings	GloVe 300
Layers	2
BatchNorm	present
Activation	ReLU
Dropout	0.95
POS embed. dimension	30
Swapped words treshold	30%
Replaced words treshold	10%
Stopwords removal	Yes
Optimizer	Adam lr=0.001

Table 3: WiC classifier configuration

Model	Accuracy (%)
Bi-LSTM	76
Bi-LSTM + Cosine	72.5
Bi-LSTM + NLTK POS	85.1

Table 4: Context Encoder performance

Model	Accuracy (%)
Baseline GloVe50	68.5
Cosine similarity from CE	71.17
CE + MLP	73.6
CE + MLP + POS	**73.8**
CE + MLP + Multi-learning	73.5
CE + MLP + NLTK POS	70.1

Table 5: Performance of the original WiC problem

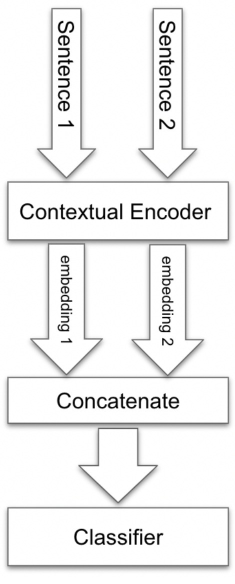

Figure 1: Model architecture

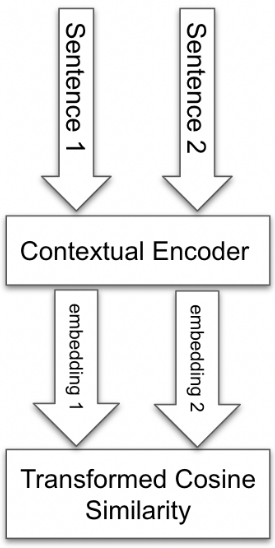

Figure 2: Model with transformed cosine similarity (TCS) classifier instead of multi-layer perceptron. Model is reduced in terms of parameters because TCS does not contain any.

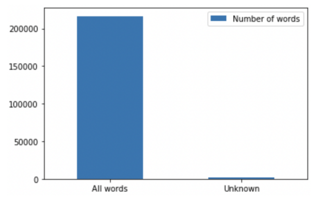

Figure 3: Comparison of number of all words to OOV words

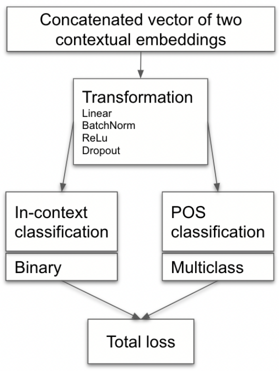

Figure 4: Classifier with an additional loss for POS classification

References

Shahul ES. 2021. Data augmentation in nlp: Best practices from a kaggle master.

Chetna Khanna. 2010. Text pre-processing: Stop words removal using different libraries.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Alessandro Raganato, Claudio Delli Bovi, and Roberto Navigli. 2017. Neural sequence learning models for word sense disambiguation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1156–1167, Copenhagen, Denmark. Association for Computational Linguistics.

Dayu Yuan, Julian Richardson, Ryan Doherty, Colin Evans, and Eric Altendorf. 2016. Semi-supervised word sense disambiguation with neural models.

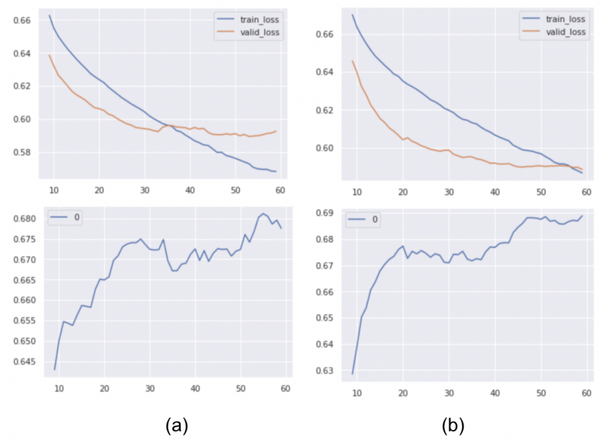

Figure 5: Comparison of two training procedures of the baseline: (a) without random replacement and (b) with 20% threshold of replacement. Plots are moving averages with window size of 20