Natural Language Processing - Homework 1
Word-in-Context Disambiguation

Lorenzo Nicoletti - 1797464
Università di Roma ”La Sapienza”
nicoletti.1797464@studenti.uniroma1.it

Abstract

Word-in-Context Disambiguation (WiCD) is a Natural Language Processing task characterized by a very high difficulty, where the objective is to predict whether a specific target word occurring in a pair of sentences shares the same context or not.

1 Introduction

The following sections of this report will explain the proposed approaches with their advantages and disadvantages and their preprocessing stage with all the needed operations on data in order to facilitate the model predictions. It will discuss several implementations of Neural Network architectures by analyzing their results in terms of the most used evaluation metrics and it will describe how some of the best practices in the Machine Learning field have lead to a boost of performances.

2 Related work

The assigned work has required a preliminary phase of analysis of the task and designing of the most feasible approaches to complete it. WiCD is a very difficult task even for state-of-the-art networks and its difficulty significantly grows when contextualized embeddings or transformers are not allowed. Therefore, the best way to face the problem is to rely on some of the most common models used nowadays, namely Recurrent Neural Network (RNN) and its variations, Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU). Moreover, the usage of a pre-trained word embeddings will help the construction of the vocabulary to train the network; in particular, the succeeding experiments will exploit *GloVe* (Pennington et al., 2014), trained on Wikipedia 2014 and English Gigaword Fifth Edition, with a total of 6B tokens, 400k vocabs and 300-dimensional vectors.

3 Method

The adopted method has been structured following two different approaches including also different preprocessing and different model architectures.

3.1 Approach 1: Word-level

This approach is very simple but at the same time also quite efficient and can be considered as a baseline on which more articulated models will be built. The Neural Network adopted in this section is very basic and straightforward: it is composed by only three Linear layers organized in a sequential structure, where the output of a layer is the input of the next one. The most interesting aspect of this stage is the preprocessing adapted on the given data for training and validation.

Simple preprocessing: The very first manipulation of data is based on the fact that the model input is mainly composed by a target word, the lemma, and a pair of sentences where it occurs. Exploiting Glove pre-trained embeddings, the two sentences have been split into tokens and associated to their latent vector representations, that have been averaged and finally concatenated in a (2 x 300)-dimensional vector.

Context Weight Estimation (CWE): This second and more advanced preprocessing step is directly derived from the first one described above. Since the task concerns context disambiguation with the only possibility to use uncontextualized embeddings, the idea of trying to estimate the ’context weight’ was born to supply this lack. The intuition is very simple: if a target word in two sentences shares the same context, then the two most related words in these two sentences (one per sentence) may be related to each other too. The relation between words is measured with the *cosine similarity* (Lahitani et al., 2016). This distance will be first computed between the target and all the other words in the pair and then the two ones with the highest scores will be used to compute another cosine similarity and this amount will be the 'weight' of the context to be multiplied by the first concatenation. This estimation will improve model performances, as the following parts of the report will analyze. Moreover, this algorithm works better when the target occurs in its 'lemma-form', therefore its substitution with the lemma has been also tested with quite good improvements.

3.2 Approach 2: Sequence Encoding

The second approach makes a huge use of advanced recurrent nets to encode sequences of words, namely the aforementioned RNN, LSTM and GRU, and, in particular, their *bidirectional variations* (Schuster and Paliwal, 1997). The model for this approach is consistently more articulated than the previous one: it is composed by a first Embedding layers followed by a (bidirectional) recurrent layer and two final Linear layers.

Indexed representation of words: The preprocessing is focused on the creation of a dictionary of indices of words to be associated to their vector representations: an index is a unique correspondence between a word and its vectorized form. Special indices are used, namely for the padding, the unknown and the separator (between two sentences) tokens. These elements will end up in the embedding matrix to be parsed by the embedding layer of the network.

4 Experiments

This work has required a wide section of experiments useful to tune hyperparameters and identify the most performing models among the implemented approaches. They have exploited GloVe embeddings and some Machine Learning best practises, mainly *Dropout* (Srivastava et al., 2014) and *Early Stopping* (Prechelt, 1998), have been adopted. All their characteristics have been grouped into tables and informative plots that will now be discussed and analyzed.

5 Results

All the models have been trained with *Adam optimizer* (Kingma and Ba, 2014), while data have been preprocess with a filtering list of non-relevant words to reduce the amount of not required information. Starting from the linear baseline and the chosen parameters configuration grouped in table 1, the results testify how the application of CWE algorithm has slightly improved performances (even better with lemmatization of target word). The standard baseline has higher recall and F1 for class 'True' but CWE has introduces very good improvements also for class 'False', as table 2 reports. The final model with lemmatization is, indeed, the best one for this first approach. Finally, Early Stopping has been tested without significantly advantages and therefore, it has been discarded.

The comparison among recurrent nets has registered that unidirectional model are not so efficient, while, on the other hand, bi-LSTM and bi-GRU are the most performing models. In only ten epochs, they managed to achieve quite good performances (table 3). Moreover, the Drop Out regularization has improved performances of 2% at least. In the per-class analysis of table 4, bi-GRU + DO is the overall best model for the second approach.

Word-level models are clearly the ones with higher performances; however, figure 1, compared to figure 2, shows that their training trends are quite irregular: the validation losses diverge, together with overfitting from about 200 epochs (model weights have been saved right before), while sequence encoding models report smoother plots. But the most interesting comparison is inducted by the related confusion matrices of figure 3: CWE model is the only managing to increase its accuracy for the class 'False', while the other tree models tend to predict more 'True'. In particular, CWE + Lemm. is the more accurate model but bi-LSTM and mostly bi-GRU are able to better identify the same context of a target word. Finally, figure 4 sums up all the conclusions by visualizing the correspondent *ROC curves and their AUC scores* (Bradley, 1997): as expected, the best model of Approach 1 (CWE + Lemm.) is preferred to bi-GRU + DO, the best model of Approach 2.

6 Conclusions

This work testifies how WiCD is difficult with the given limitations. The proposed approaches provide simple but also quite efficient implementations, with boosting of performances thanks to pretrained embeddings and best practises, like regularization or a filtered-words list. Their strenghts and weaknesses have been evaluated with common metrics and plots to identify the best model, with final encouraging achievements in terms of accuracy.

References

Andrew P. Bradley. 1997. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30:1145–1159.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *International Conference for Learning Representations*.

Alfirna Rizqi Lahitani, Adhistya Erna Permanasari, and Noor Akhmad Setiawan. 2016. Cosine similarity to determine similarity measure: Study case in online essay assessment. 4th International Conference on Cyber and IT Service Management:1–6.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP):1532–1543.

Lutz Prechelt. 1998. Early stopping - but when? Neural Networks: Tricks of the Trade:55–69.

Mike Schuster and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing:2673–2681.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research (JMLR):1929–1958.

A Tables

All the models in this section have exploited GloVe pre-trained embeddings.

Model	Hidden size	LR	Epochs	Accuracy
Baseline	128	$1e-4$	500	0.6677
Baseline + CWE	128	$1e-4$	500	0.6720
Baseline + CWE + Lemm.	128	$1e-4$	500	**0.6770**

Table 1: Tested models for the Approach 1, with fixed hidden size and learning rate (LR). CWE stands for Context Weight Estimation and Lemm. for Lemmatization of the target word.

Model	Class	Prec.	Recall	F1
Baseline	1	0.64	**0.75**	**0.69**
	0	**0.70**	0.59	0.64
	avg	0.67	0.67	0.67
Baseline + CWE	1	**0.69**	0.62	0.65
	0	0.66	**0.72**	**0.69**
	avg	0.67	0.67	0.67
Baseline + CWE + Lemm.	1	**0.69**	0.63	0.66
	0	0.66	**0.72**	**0.69**
	avg	**0.68**	**0.68**	**0.68**

Table 2: Precision, Recall and F1-score for the models of Approach 1 with respect to each class and to the average.

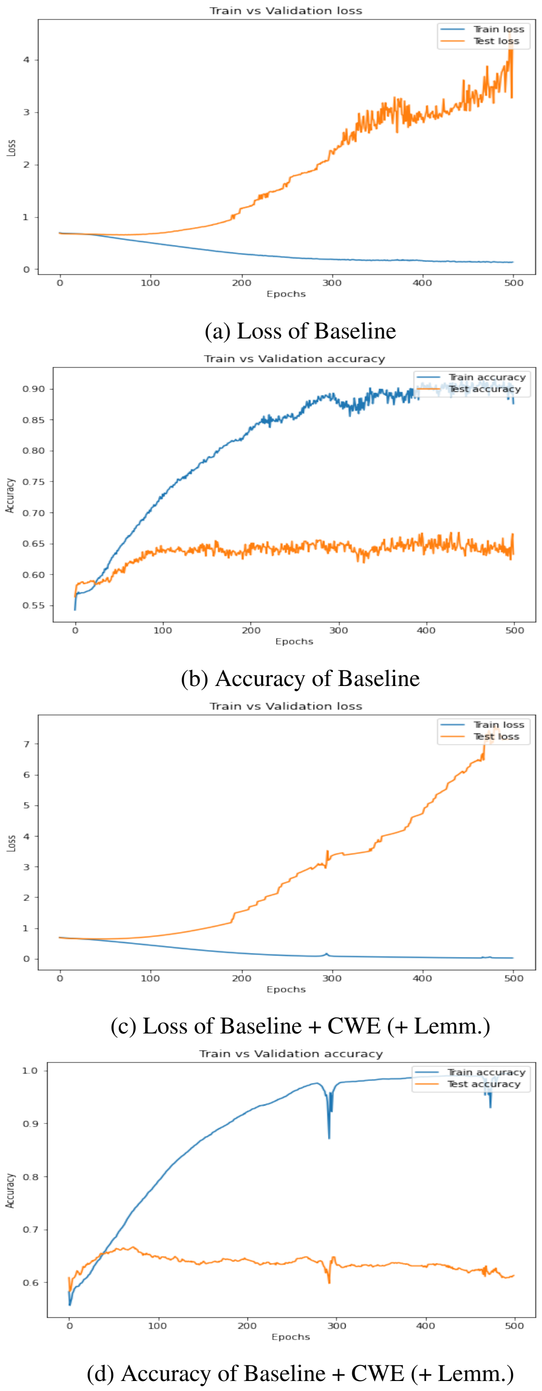
Model	Hidden size	Rec. layers	LR	DO	Epochs	Accuracy
RNN	128	1	$1e-4$	-	10	0.5220
LSTM	128	1	$1e-4$	-	10	0.5310
GRU	128	1	$1e-4$	-	10	0.5310
bi-RNN	128	2	$1e-4$	-	10	0.5340
bi-LSTM	128	2	$1e-4$	-	10	0.5990
bi-LSTM	128	2	$1e-4$	0.2	10	**0.6170**
bi-GRU	128	2	$1e-4$	-	10	0.6010
bi-GRU	128	2	$1e-4$	0.2	10	**0.6270**

Table 3: Tested models for the Approach 2, with fixed hidden size and learning rate (LR) + varying bidirectional structure and Drop Out (DO) regularization. The models are denoted by only their recurrent layer for simplicity but their overall structures comprehend embedding and linear layers, as explained in *Section 3.2*.

Model	Class	Prec.	Recall	F1
bi-LSTM + DO	1	0.60	**0.72**	0.65
	0	0.65	0.51	0.57
	avg	0.62	0.62	0.61
bi-GRU + DO	1	**0.61**	0.70	0.65
	0	0.65	**0.55**	**0.60**
	avg	**0.63**	**0.63**	**0.62**

Table 4: Precision, Recall and F1-score for the best two models of Approach 2 with respect to each class and to the average.

B Training and Validation plots



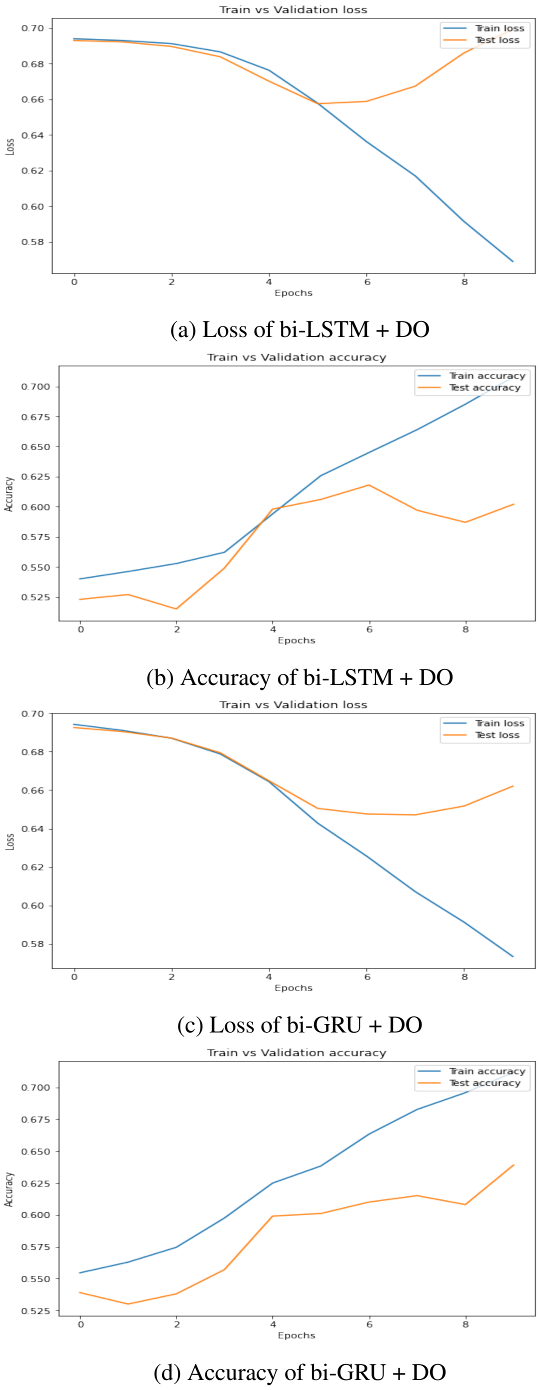
(a) Loss of Baseline

(b) Accuracy of Baseline

(c) Loss of Baseline + CWE (+ Lemm.)

(d) Accuracy of Baseline + CWE (+ Lemm.)

Figure 1: Training and validation plots for Approach 1. CWE with or without Lemm. plots are actually the same and only one (with Lemm.) has been reported.



(a) Loss of bi-LSTM + DO

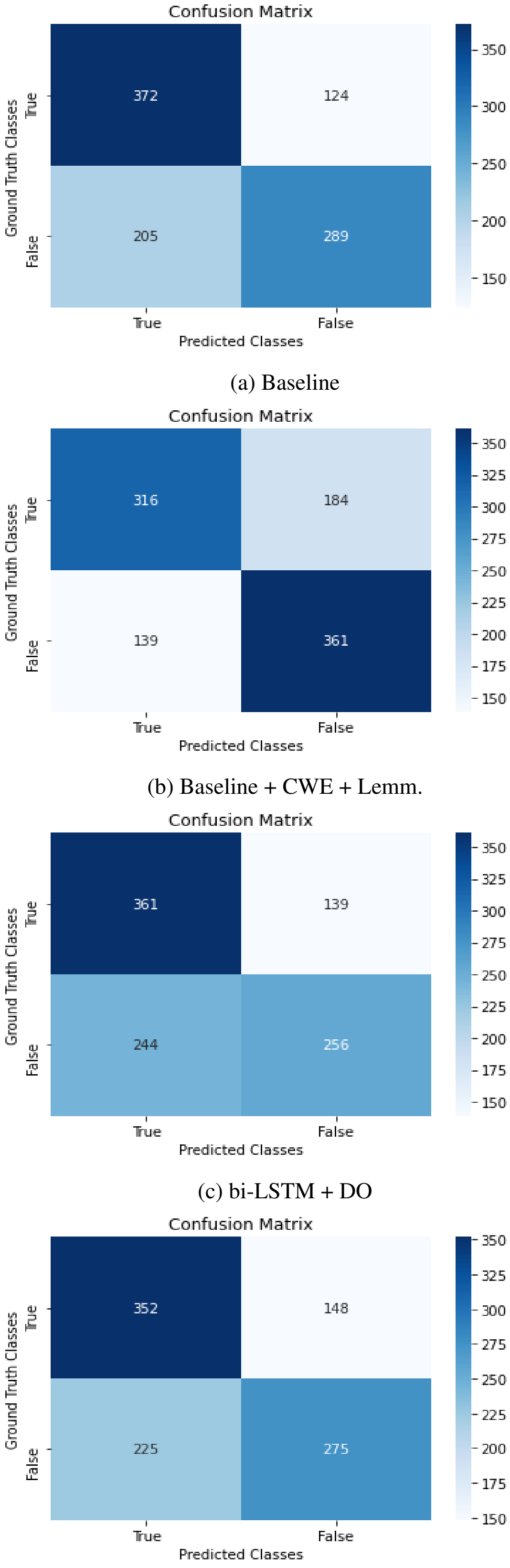
(b) Accuracy of bi-LSTM + DO

(c) Loss of bi-GRU + DO

(d) Accuracy of bi-GRU + DO

Figure 2: Training and validation plots for the best two models of Approach 2.

C Confusion Matrices



(a) Baseline

(b) Baseline + CWE + Lemm.

(c) bi-LSTM + DO

(d) bi-GRU + DO

Figure 3: Confusion matrices of the four models.

D Comparative plot: ROC curve and AUC score

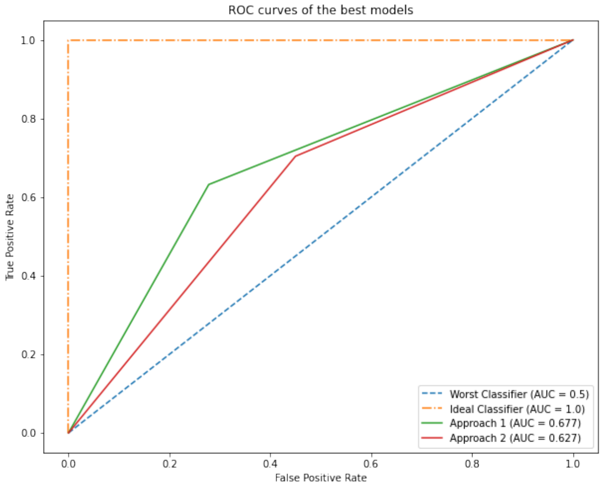


Figure 4: ROC curves of Baseline + CWE + Lemm. (Approach 1) and bi-GRU + DO (Approach 2) compared to the worst/random (dotted blue line) and ideal (dotted orange line) classifiers. In the legend on the bottom-right corner, the associated AUC scores are reported.