INTRODUCTION

The task of Word Sense Disambiguation consists of identifying which sense of a word is used in a sentence. The main problem is that words might have more than one meaning, which can be determined only by examining the context where that word is used and the Part-of-speech (POS) tagging of that ambiguous word. Many different approaches have been adopted to develop WSD systems: **supervised methods** [1], where WSD is configured as a classification problem, **unsupervised methods** [2] based on the idea that the same sense of a word will have similar neighbouring words and **knowledge-based methods** [3] based on the use of a knowledge resource to assign the appropriate senses to a target words in the context. This project implements the supervised classification task and aims to develop English multi-inventory Word Sense Disambiguation systems, which are able to perform fine-grained (BabelNet IDs) and coarse-grained WSD (WordNet Domains and Lexicographer's IDs).

PRE-PROCESSING

The input and output vocabularies are created from the training set: the input vocabulary (which includes the tokens for the padding and for the unknown words) is built by giving an ID to every training word. Furthermore, all non-instance words which occur less than 3 times are considered as unknown words. The output vocabularies (including the tokens for the padding, for the unknown words and the unseen instance words) are generated from the training gold file (BabelNet IDs, WordNet Domains and Lexicographer's IDs vocabularies) and the training xml file (POS tagging vocabularies). The candidate outputs (BabelNet IDs, WordNet Domains and Lexicographer's IDs), for all the instance words are created, consisting of only those outputs of a lemma encountered in the training set given the part-of-speech tag. In the end, the sentences are tokenized, split and padded (each sentence has a fixed length equal to 260) before feeding them to the WSD models.

MODEL

Two different models are implemented to perform fine-grained and coarse-grained WSD; although the models perform different tasks, they share the same architecture, as shown in Fig.1, whose main components are:

- ELMO embedding layer: Elmo representation is a new type of deep contextualized word representation [4], which models complex characteristics of word use and how these uses vary across linguistic contexts. Instead of using a fixed embedding vectors built without considering the context, Elmo is dynamic, meaning that its embeddings change depending on the context, indeed it looks at the entire sentence before assigning an embedding to each word even when the word is the same.
- One layer of bi-directional LSTM is used to exploit information from the whole sequence and encode it in the hidden states. The output vector is obtained by concatenating the outputs of the forward and backward vectors.
- The attention mechanism [5] is used to create a context vector from all the hidden states (i.e. global attention [6]). It captures relevant information to help the prediction of the current target word, because certain elements can be more discriminative than others in predicting the output label. Then the context vector and the output vector are concatenated.
- Multi-task learning: the main idea is to improve the performance of the task-specific models by solving multiple tasks at the same time, while exploiting commonalities and differences across them. The two implemented models have the same architecture with multiple loss functions and a shared representation as shown in Fig.1. In the fine-grained model, the auxiliary tasks include the POS tagging, the coarse-grained semantic domains (WordNet domains) and the ontological classes (LexNames). The POS tagging helps in dealing with cross-POS lexical ambiguities, whereas the coarse-grained predictions help the model generalize, especially for those senses less covered at training time. In the coarse-grained model, the auxiliary tasks are the coarse-grained representation and the POS tagging; the fine-grained task is not included, because it does not add any useful information to help the model generalize for a better score. A joint loss L_{joint} is added and optimized in both models, and it is defined as the sum of the individual-tasks losses. It is used to preserve the independence of the task-specific functions, preventing the model from becoming biased towards a specific task. In the fine-grained model, the joint loss is defined as

  $$L_{joint} = L_{POS} + L_{WN} + L_{LEX} + L_{FINE}$$

  while in the coarse-grained model is defined as

$$L_{joint} = L_{POS} + L_{WN} + L_{LEX}.$$

EXPERIMENTAL DATASETS

The two WordNet sense-annotated corpora used to train the models are:

1. *OMSTI:* A large corpus automatically annotated with senses from the WordNet 3.0 inventory.
2. *SemCor:* A manually sense-annotated corpus divided into 352 documents for a total of 226.040 sense annotations, whose quality is expected to be higher than the quality of OMSTI, because it was manually built.

Five datasets from the Senseval and SemEval competitions are considered as the evaluation framework:

1. *SemEval-07 task 17:* it is the smallest dataset, containing 455 sense annotations for noun and verbs only.
2. *SemEval-13 task 12:* this dataset includes thirteen documents from various domain, including 1644 sense annotations, although only nouns are considered.
3. *SemEval-15 task 13:* it consists of 1022 sense annotations from three heterogeneous domains (biomedical, mathematics/computing and social issues).
4. *Senseval-2:* it consists of 2282 sense annotations, including nouns, verbs, adverbs and adjectives.
5. *Senseval-3 task 1:* it consists of three documents from three different domains (editorial, news story and fiction).

In Raganato *et al.* [7], the ambiguity level[1] is calculated, showing that *OMSTI*, despite being constructed automatically, contains a high coverage of ambiguous words; whereas *SemEval-07* contains the highest ambiguity level among all evaluation datasets, therefore this dataset is chosen as development set.

EVALUATION

The metric used to evaluate the performance of the models is the F1 scores, which is defined as follows:

$$F_1 = 2\frac{PR}{P+R},$$

where P is the precision and R is the recall[2]. A backoff strategy is implemented to provide an answer for the lemmas which are not in the training set, this strategy is the MFS (Most Frequent Sense) which returns the predominant sense of a lemma, therefore the F1 score becomes

$$F_1 = P = R.$$

SELF-COMPARISON

The self-comparison is studied by fixing the hyperparameters of the models (showed in Table 4 and inspired by Raganato *et al.* [1]), and by training the model with the same number of sentences (576,000 sentences). Therefore, the models without Elmo are trained using a batch size of 64 for 30 epochs with 300 iterations, whereas the models with Elmo are trained using a batch size of 16 for 30 epochs with 1200 iterations. The different batch size is due to the complexity and the size of word embeddings of Elmo representation (showed in Table 4), indeed the introduction of Elmo embeddings makes the final architecture much more complex. The Table 1, 2 and 3 show the improvement obtained by including the attention layer, the multi-task learning and Elmo embeddings. The inclusion of the attention layer above the BiLSTM model with a single output enhances the performance on the fine-grained task, whereas for both coarse-grained predictions, the improvement is very small (LexNames) or null (WordNet domains).
The multi-task learning increases the final score for both models: the introduction of the LexNames and the WordNet domains as additional predictions in the fine-grained model improves the final score (highlighting that the coarse-grained predictions can boost the fine-grained), likewise the score for a coarse-grained prediction increases by the introduction of the other coarse-grained representation as additional task. Using the contextualized word representation introduced by Elmo, the score is increased for each prediction, especially for the LexNames (where from a F1 score of 75.8 it reaches a F1 score of 80.2). The POS prediction improves all the coarse-grained results especially for the WordNet domains which final score increases of 0.6, whereas for the fine-grained result the final score with the POS tagging is lower than the final score without it, as shown in Table 1.

[1] The ambiguity level is computed as the total number of candidates senses divided by the number of sense annotations.
[2] The precision is the number of correct answers over the number of answers the system gives, whereas the recall is the number of correct answers over the total correct answers the system should give.

RESULTS

After tuning the hyperparameters as shown in Table 4, the best models reach respectively a F1 score of 65.2 for the fine-grained prediction, 86.3 for the coarse-grained WordNet domains prediction and 80.6 for the coarse-grained LexNames prediction on the concatenation of the 5 development datasets. The fine-grained prediction is boosted by introducing a multi-task learning where the POS tagging, Wordnet domains and LexNames IDs are predicted as well. This is justifiable by the fact that these auxiliary tasks help the model generalize and efficiently disambiguate words by using the POS tagging to distinguish words based on the lexical domain (e.g. book, bank) and the coarse-grained information, which introduces semantic and ontological information (LexNames IDs and WordNet domains, respectively) to disambiguate words. On the other hand, the fine-grained task does not help in predicting the coarse-grained representation (as shown in Table 2 and Table 3). One possible explanation can be the fact that each LexNames and WordNet domain can be mapped to more than one fine-grained ID, thus the fine-grained task does not help the coarse-grained model generalize but only ends up adding noise in the model, thereby affecting the final score. The predictions of each coarse-grained label category achieve better results when using POS tagging and the other coarse-grained label category as auxiliary tasks. The POS tagging adds lexical information and two different coarse-grained labels introduce semantic and ontological domains, useful information to disambiguate words in a sentence. The model is able to perform better than the individual-coarse-grained-task model and the model supported by the fine-grained prediction, although the mapping between the WordNet domain and the LexNames is not one-to-one, as the mapping between the BabelNet IDs and both coarse-grained representations is indeed less imbalanced, and, moreover, adds information from the semantic and the ontological point of views to the same general representation of the words (coarse-grained).

CONCLUSION

In this work, two WSD models are implemented: one exploits the coarse-grained and POS tagging information to improve the fine-grained prediction, while the other model is trained to disambiguate words in the coarse-grained task by exploiting the WordNet domains, LexNames IDs and POS tagging information. In both models a joint loss defined as the sum of the individual-task losses is introduced to prevent the models from being biased towards a specific task. A possible improvement can be done by introducing a weighted sum instead of a simple sum of the losses in the multi-task learning, through the introduction of some weights for the individual loss in the joint loss L_{joint} to give more importance to specific components.

REFERENCES

[1] Alessandro Raganato, Claudio Delli Bovi, Roberto Navigli: Neural Sequence Learning Models for Word Sense Disambiguation. Proceedings of EACL 2017, Copenhagen, Denmark.

[2] Eneko Agirre and Aitor Soroa. 2009. Personalizing PageRank for word sense disambiguation. In Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics (EACL '09). Association for Computational Linguistics, Stroudsburg, PA, USA, 33-41.

[3] Michael Lesk. 1986. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In Proceedings of the 5th annual international conference on Systems documentation (SIGDOC '86), Virginia DeBuys (Ed.). ACM, New York, NY, USA, 24-26. DOI: https://doi.org/10.1145/318723.318728.

[4] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, Luke Zettlemoyer: Deep Contextualized Word Representations. NAACL-HLT 2018: 2227-2237.

[5] Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, Bo Xu: Attention-Based Bidirectional Long Short-Term Memory Networks for Relation Classification. ACL (2) 2016.

[6] Thang Luong, Hieu Pham, Christopher D. Manning: Effective Approaches to Attention-based Neural Machine Translation. EMNLP 2015: 1412-1421.

[7] Alessandro Raganato, Jose Camacho-Collados and Roberto Navigli: Word Sense Disambiguation: A Unified Evaluation Framework and Empirical Comparison. Proceedings of EACL 2017, Valencia, Spain.

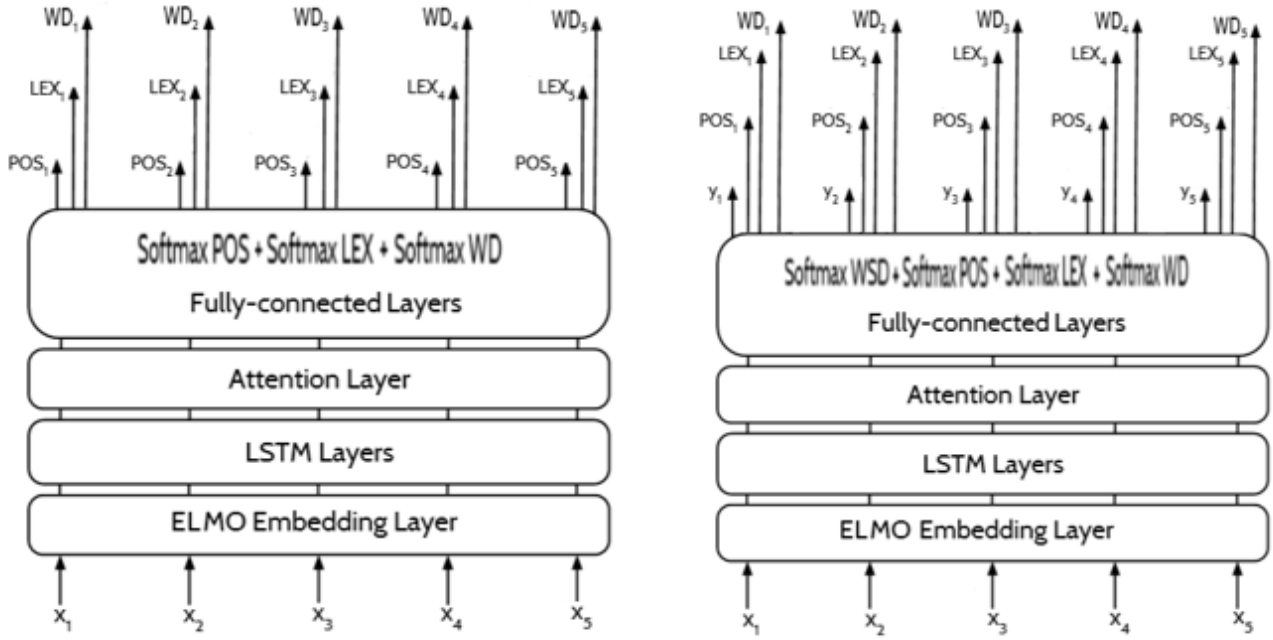

Figure 1: The right picture shows the architecture of the fine-grained model, whereas the left picture shows the architecture of the coarse-grained model.

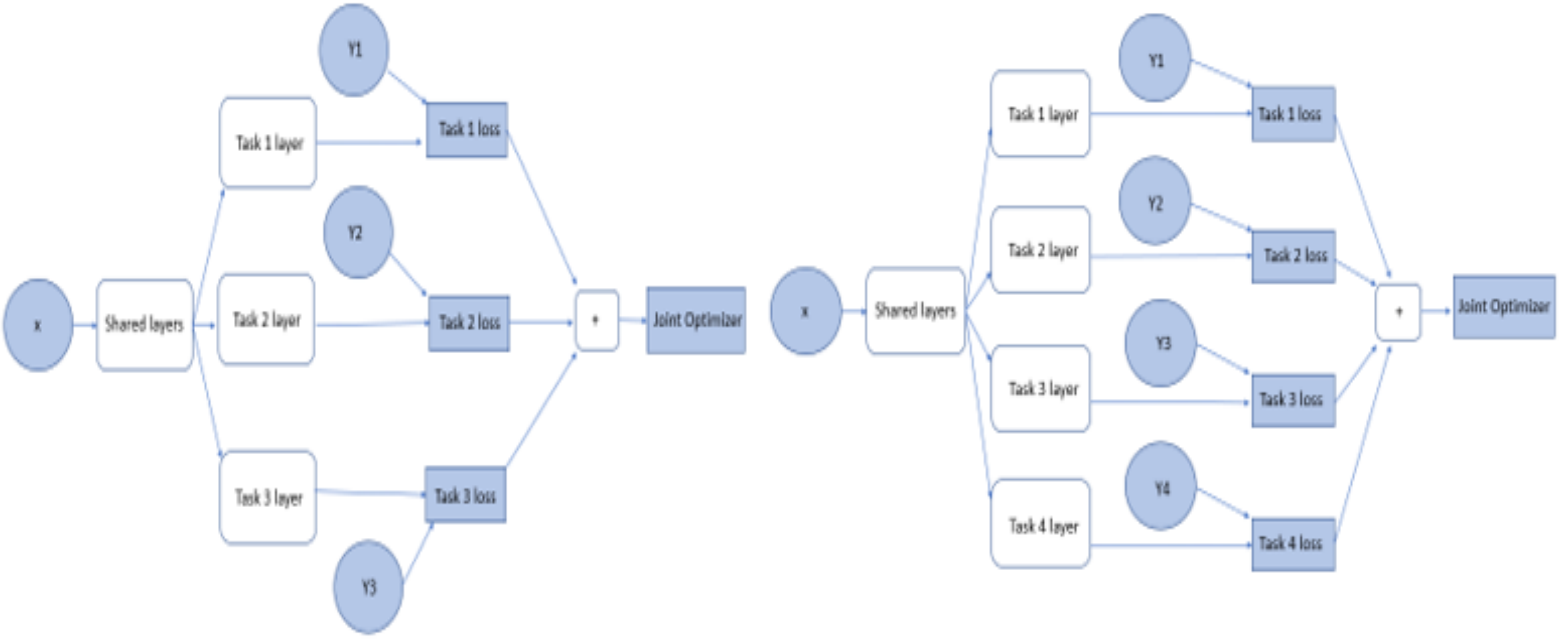

Figure 2: Structures of the loss used in the two models (the right picture shows the loss structure of the fine-grained model and the left picture shows the loss structure of the coarse-grained).

Model	SemEval2007	SemEval2013	SemEval2015	Senseval2	Senseval3	Concatenate
BiLSTM	45.9	59.5	53.1	63	55.5	57.9
BiLSTM + ATT	47.5	60.2	55.6	62.8	56.9	58.7
BiLSTM + ATT + LEX	49.7	60.2	57.6	64.9	59	60.3
BiLSTM + ATT + LEX + WNDOMAIN	46.6	61.6	57.4	65.2	59.8	60.7
BiLSTM + ATT + LEX + WNDOMAIN + ELMO	48.6	62.2	59.3	66.5	60.8	61.9
BiLSTM + ATT + LEX + WNDOMAIN + POS + ELMO	48.8	61.1	59.3	65.2	62.6	61.8
BiLSTM + ATT + LEX + WNDOMAIN + POS + ELMO (BEST PARAMETERS)	**55.2**	**63.8**	**59.6**	**69.7**	**66.3**	**65.2**

Table 1: F1 scores for the BabelNet ids predictions (fine-grained), obtained by using fixed parameters.

Model	SemEval2007	SemEval2013	SemEval2015	Senseval2	Senseval3	Concatenate
BILSTM	84.6	76	81.4	88.7	81.9	82.8
BILSTM + ATT	85.9	75.5	82.2	88.6	81.8	82.8
BILSTM + ATT + FINE + LEX	84.8	76.6	82.4	88.3	82.9	83.2
BILSTM + ATT + LEX	85.5	76.5	82.2	89.6	84.8	84.1
BILSTM + ATT + LEX + ELMO	87.7	78	83.8	**91.3**	86.3	85.7
BILSTM + ATT + LEX + POS + ELMO (BEST PARAMETERS)	**88.8**	**78.8**	**84**	91.2	**87.6**	**86.3**

Table 2: F1 scores for the WordNet Domain predictions (Coarse-grained), obtained by using fixed parameters.

Model	SemEval2007	SemEval2013	SemEval2015	Senseval2	Senseval3	Concatenate
BILSTM	64.8	71.5	74.3	80.9	71.5	74.4
BILSTM + ATT	65.9	71.6	75.4	80.2	73.4	74.9
BILSTM + ATT + FINE + WNDOMAIN	64.4	70.5	74.8	81.6	74.7	75.3
BILSTM + ATT + WNDOMAIN	68.1	72.1	75.3	81.2	74.8	75.8
BILSTM + ATT + WNDOMAIN + ELMO	71.9	76	78.8	85.2	**80.6**	80.2
BILSTM + ATT + WNDOMAIN + POS + ELMO (BEST PARAMETERS)	**73**	**76.3**	**80.3**	**85.5**	80.3	**80.6**

Table 3: F1 scores for the Lexname predictions (Coarse-grained), obtained by using fixed parameters.

Hyper-Parameters	Values			Self-comparison parameters		Best Parameters (Fine-grained)	Best Parameters (Coarse-grained LexNames)	Best Parameters (Coarse-grained WN Domain)
Optimizer	Adam	Adadelta		Adadelta		Adam	Adam	Adam
Learning rate	0.001	(0.5, 1)		1		0.001	0.001	0.001
Epochs	(30, 40)	30		30		40	30	30
Hidden size	256	(128, 256)		128		256	256	256
Elmo Embedding	Yes	Yes	No	Yes	No	Yes	Yes	Yes
Embedding size	1024	1024	300	1024	300	1024	1024	1024
Batch size	16	16	64	16	64	16	16	16
Iterations	1200	1200	300	1200	300	1200	1200	1200
Num. layers	1			1		1	1	1
Keep Prob	0.9			0.9		0.9	0.9	0.9

Table 4: Set of values used during the tuning phase, and the best parameters for the fine-grained and coarse-grained model.

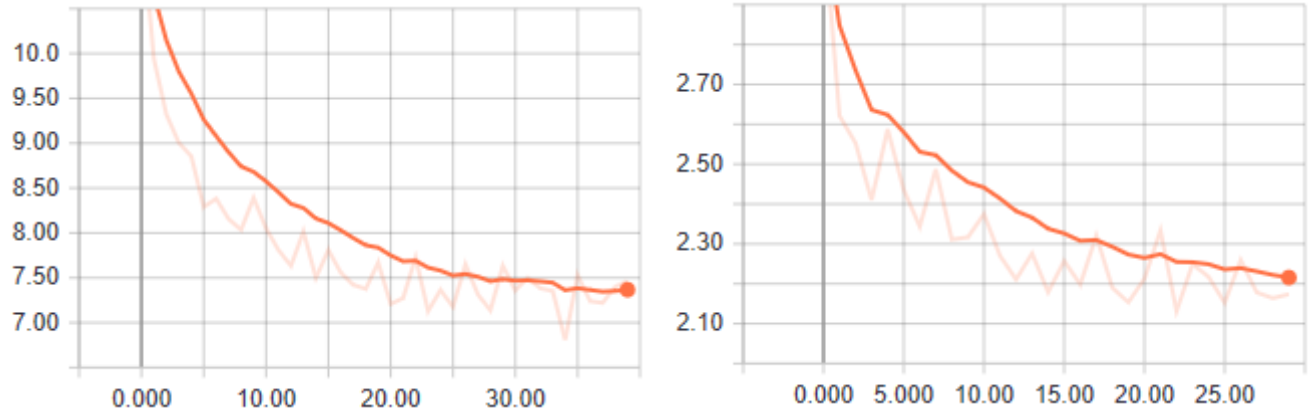

Figure 3: The joint losses on the development set: the left plot is the joint loss of the coarse-grained model; the right plot is the joint loss of the fine-grained model.