Three Classifiers, One Goal. The Influence of Word Embeddings on the Semantic Role Labeling Task.

Nils-Jonathan Schaller
Vrije Universiteit Amsterdam
Humanities Research: Linguistics: Human Language Technologies
schaller.nj@gmail.com

Abstract

Word embeddings have given a strong performance boost in a variety of natural language processing task. To explore the influence of word embeddings on semantic role labeling (SRL) for the dutch language in this paper, three models of support-vector machine (SVM) classifiers are constructed and evaluated. The first model is based on features given by annotation and the second model is, in contrast, a classifier based purely on word embeddings and the third combines both models. The results contradict the expectation of the comparison, that the second model outranks the first one, whereas the combination should get the best performance overall. Instead, the feature based model had the best performance. To prevent this, a change in the selection of analysed tokens is suggested.

1 Introduction

The performance of semantic role labeling can potentially help in solving a variety of natural language processing tasks. As there is still a lack of research on other languages than English, this paper focusses on the classification of semantic roles in a dutch corpus. The purpose is, to identify features that improve SRL. This is tested on the basis of three training models. The first of these models extracts a variety of syntactic features, whereas the second model focusses only on word embeddings. A third model tests, if a combination would improve or impair the performance for the task. Due to the success of using word embeddings in NLP tasks, the assumption is, that the second model will result in a higher performance than the first, and that a combination of both variants improves the classification even further.

2 Related Work

De Clercq et al (2015) describe the evaluation of an automatic SRL process using cross-domain texts while focusing whether varying the genre and amount of training data improves the performance. One of the findings was, that having a large data set for training is a necessity for a good semantic role labeler, but that adding a small corpus of domain specific texts improves the results. In the experiment 1/6th of the training corpus were in-domain texts, which delivered the best results.

The experiments reveal that training on large data sets is necessary but that including genre-specific training material is also crucial to optimize classification. (de Clercq et al., 2015)

The labeling was based on propbank rules, another finding was, that higher numbered arguments (Arg3, Arg4 etc.) were more difficult to label. As the predicates were mapped onto English Propbank, sentences with multiple possible translations turned out to be more problematic. In general, Propbanks crosslingual validity was confirmed. The corpus of 500.000 words consists of six distinct genres. It was annotated based on dependency syntax.

Semantic Role Labeling is the task of extracting the information *who* did *what* to *whom* and possibly *where* and *when*. To do so, parts of a text are labeled for different semantic roles. The roles can be described as the participants of an event, the active and passive participants, the theme, cause etc. If those roles are identified, it could improve information extraction and therefore also improve other NLP tasks.
Examples for semantic roles:
AGENT: The volitional causer of an event. *The waiter* spilled the soup.
EXPERIENCER: The experiencer of an event.

John has a headache.
FORCE: The non-volitional causer of the event. *The wind* blows debris from the mall into our yards. Examples from: (Jurafsky and Martin, 2018)

A major work in annotating semantic roles has been done by Palmer et al. (2005) The PropBank annotates predicates semantic roles by focusing on the argument structure of verbs and provides a completely annotated corpus. The underlying work is based on studies about the linking between semantic roles and syntactic realization. PropBank defines semantic roles on a verb by verb basis, due to the problematic of defining universal semantic roles. Each predicate has a number of conditional and sometimes additional arguments, that are always starting by zero (Arg0, Arg1, Arg2 etc.). Arg0 is in general close to a prototypical agent, while Arg1 is close to a prototypical patient or theme. Arguments higher than that cannot be labeled consistently. Those arguments form a roleset for each predicate or a frameset, when associated with syntactic frames. Depending on the polysemy of the verb, a predicate can have multiple framesets. The total of the framesets for a verb are refered to as a frames file. (Palmer et al., 2005)

3 Methodology

3.1 Data

The corpus for the SRL classification is developed by De Clercq et al (2015) for the dutch language and consists of 500.000 words of written dutch text from various genres that have been manually labeled following the PropBank scheme (Palmer et al., 2005) and based on that further 500.000 automatically labeled words, e.g. 1 million words in total.

3.2 SVM

A support vector machine is a supervised machine learning system, usually used for binary classification. It can also be applied on multi class classification tasks, therefore it is sufficient for SRL. The textual features are converted into vectors.

3.3 Feature based model

The data consists of NAF files and is already labeled for syntactical information. Each semantic role relates to a sentence chunk which consists of a span of different lemmas. The chunk is extracted and, using the spacy module, a dependency parser chooses the root. For this root, a number of features is extracted: the lemma of the root, the part of speech tag, the morphofeat and the lemma of the predicate belonging to this entity of the semantic role. The features were added into a dictionary related to the semantic role label of the root. In a following step, the textual features were transferred into a vectoral representation for the SVM.

3.4 Word Embedding Model

The word embedding model also extracts information based on the root of each semantic role. For each root the word embeddings are created with the gensim module which creates a vector based on word2vec data. Due to computational reasons, the 160-dimensional word embedding version ”Combined” from Tulkens et al. (2016) was selected. This resulted in 160 vectors which were stored as 160 features relating to the SRL of the root and fed into the SVM.

3.5 Combined Model

To examine, how much both models gain from each other, a third model was created. In here both the syntactic features and the word embeddings are saved as features for the semantic role labeling. The expectation is, that the combination will result in a higher performance

4 Results and error analysis

4.1 Results

In the experiment three models were created. One with and one without word embeddings and finally a combinated model. The feature based model (Table 2) had a F1 score of 0.56 on the test set, which is just slightly better than chance. It succeeded especially well in classifying some of the modifier arguments ArgM-DIS, ArgM-NEG and ArgM-REC. The word embeddings based module (Table 1) got a F1 score of 0.49 on the test set. Both modules did a similar low prediction with the most frequent arguments Arg0 and Arg1, whereas the feature based version was slightly better and was also able to classify arguments, that appear very rarely. In contrary, the word embedding model had difficulties in classify all arguments, that occurred very rarely. There is no clear pattern for a relation of the frequency of an argument and the performance in classifying that

Table 1: SVM with word embeddings.

argument	prec	recall	f1	supp.
Arg0	0.58	0.55	0.56	1078
Arg1	0.47	0.81	0.60	2328
Arg2	0.22	0.00	0.01	635
Arg3	0.00	0.00	0.00	36
Arg4	0.00	0.00	0.00	41
Arg5	0.00	0.00	0.00	1
ArgM-ADV	0.62	0.21	0.32	423
ArgM-CAU	0.90	0.21	0.34	126
ArgM-DIR	0.00	0.00	0.00	56
ArgM-DIS	0.76	0.83	0.79	281
ArgM-EXT	0.39	0.46	0.42	70
ArgM-LOC	0.36	0.25	0.29	518
ArgM-MNR	0.56	0.25	0.35	388
ArgM-MOD	0.00	0.00	0.00	5
ArgM-NEG	0.96	0.64	0.77	179
ArgM-PNC	0.65	0.36	0.46	178
ArgM-PRD	0.88	0.12	0.22	56
ArgM-REC	0.90	0.92	0.91	120
ArgM-TMP	0.63	0.65	0.64	859
ArgM=MNR	0.00	0.00	0.00	1
SYNT	0.00	0.00	0.00	1
avg / total	**0.52**	**0.54**	**0.49**	**7380**

Table 2: feature based SVM.

argument	prec	recall	f1	supp.
Arg0	0.59	0.60	0.59	1078
Arg1	0.55	0.75	0.63	2328
Arg2	0.34	0.18	0.23	635
Arg3	0.17	0.03	0.05	36
Arg4	0.14	0.07	0.10	41
Arg5	0.00	0.00	0.00	1
ArgM-ADV	0.51	0.35	0.41	423
ArgM-CAU	0.53	0.23	0.32	126
ArgM-DIR	0.40	0.11	0.17	56
ArgM-DIS	0.81	0.83	0.82	281
ArgM-EXT	0.52	0.44	0.48	70
ArgM-LOC	0.38	0.32	0.35	518
ArgM-MNR	0.51	0.44	0.47	388
ArgM-MOD	0.67	0.40	0.50	5
ArgM-NEG	0.87	0.70	0.78	179
ArgM-PNC	0.61	0.37	0.46	178
ArgM-PRD	0.38	0.18	0.24	56
ArgM-REC	0.90	0.93	0.91	120
ArgM-TMP	0.75	0.75	0.75	859
ArgM=MNR	0.00	0.00	0.00	1
SYNT	0.00	0.00	0.00	1
avg / total	**0.56**	**0.58**	**0.56**	**7380**

argument. The combination of both models resulted in no improvement. The overall F1 score was still 0.56, the values for the particular arguments stayed the same as in model 2 or decreased slightly.

4.2 Analysis and Discussion

Contrary to the expectations, the outcome of the experiment was in favor of the feature based model. It had a better performance then the word embedding model, which did even perform slightly under chance. A combination of both versions gave no improvement to any of the results, some of the particular argument scores even decreased slightly. As to why, there are different possible explanations. First of all, the performance of the feature based model could have probably improved with adding further features, like position and distance from predicates or roots, or the consideration of using multiple values for arguments with a broad span of words and other features, as suggested in the work of De Clercq (2015). Due to the nature of this work and the timeframe, this was not possible, but could be investigated in an upfollowing project. It can be assumed, that the choice of applying the word embeddings only to the root of each semantic role chunk, plays an important role in the performance. In upfollowing experiments, this selection has to be changed and it is still unclear, which parts of a semantic role carry the most important information and which parts could potentially create noise. A selection of different factors has to be created and tested. Furthermore, due to computational power only the 160-dimensional word embedding could be used, it must also be tested, if a higher dimensional version results in a better performance.

5 Conclusion

The paper at hand did not meet the previously stated expectations. As to the reasons, it cannot be said with certainty, that the word embeddings give an inferior performance compared to the feature based model, due to the small difference in results, which were, with an F1 score of 0.56 and respectively 0.49, very low. For being able to give a more thorough error analysis, there have to be better results with a higher validity first. As mentioned before, this can probably be done through implementation of further features and trying out

Table 3: Combined SVM (features and word embeddings).

argument	**prec**	**recall**	**f1**	**supp.**
Arg0	0.60	0.62	0.61	1078
Arg1	0.56	0.73	0.63	2328
Arg2	0.31	0.18	0.23	635
Arg3	0.09	0.03	0.04	36
Arg4	0.18	0.17	0.18	41
Arg5	0.00	0.00	0.00	1
ArgM-ADV	0.52	0.35	0.42	423
ArgM-CAU	0.43	0.26	0.33	126
ArgM-DIR	0.39	0.12	0.19	56
ArgM-DIS	0.81	0.82	0.82	281
ArgM-EXT	0.49	0.43	0.46	70
ArgM-LOC	0.39	0.34	0.36	518
ArgM-MNR	0.51	0.43	0.46	388
ArgM-MOD	0.75	0.60	0.67	5
ArgM-NEG	0.85	0.70	0.77	179
ArgM-PNC	0.58	0.37	0.45	178
ArgM-PRD	0.29	0.18	0.22	56
ArgM-REC	0.90	0.92	0.91	120
ArgM-TMP	0.75	0.75	0.75	859
ArgM=MNR	0.00	0.00	0.00	1
SYNT	0.00	0.00	0.00	1
avg / total	**0.56**	**0.58**	**0.56**	**7380**

to apply the word embeddings on other tokens, for example the predicate or a span around the root or predicate. This was not possible due to the circumstances of the task, but can be applied in a further work. One interesting results is, that the combination of both models did not improve the results of the previous extracted features, but only decreased them barely for the single arguments. A comparison of the results of the other research questions in this class and a continuation of this work would give more information on how and if word embeddings improve the performance of the semantic role labeling task.

References

Orphe de Clercq, Veronique Hoste, and Paola Monachesi. 2015. Evaluating automatic cross-domain dutch semantic role annotation. 1.

Daniel Jurafsky and James H. Martin. 2018. Speech and language processing. 3rd Edition.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31:71–106.

Stphan Tulkens, Chris Emmery, and Walter Daelemans. 2016. Evaluating unsupervised dutch word embeddings as a linguistic resource. *Computation and Language (cs.CL).*

A Appendices

Python files:
model1_features.py
This file contains the feature based model.

model2_word_embeddings.py
This file contains the word embedding based model.

model3_combined.py
This file contains the combinated model of word embeddings and syntactic features.

SRL_utils.py
The utils file contains all functions for extracting all features and word embedding, as well as the training and application of the svm classifier.

Dutch Word Embeddings:
`https://github.com/clips/dutchembeddings`
A link to the used word embddings with instructions for the download. For this paper the version ”combined 160” was used and saved in a file called ”word_embeddings_160” in the same directory as the python scripts.