Aspect-based sentiment analysis

Žan Jaklič, zj8850@student.uni-lj.si
Iztok Ramovš, ir8617@student-uni.lj.si
Matjaž Mav, mm3058@student-uni.lj.si

Abstract

In this paper we discuss aspect-based sentiment analysis on Slovene news. We applied several different context extraction methods and gathered some features which we thought would be useful. We then used these features to train standard machine learning models, as well as neural networks. The models perform better than the majority classifier, but not as good as they could because of lack of information rich features and uneven class distributions.

1 Introduction

A lot of sentiment analysis has been done on short texts, e.g. on tweets, yelp reviews, amazon reviews, and some of the research has been done on longer texts like news, blog posts, etc. There is a lot of incentive to use sentiment analysis on those cases. Such techniques can, for example help companies and researchers with understanding of user opinions or filter out unimportant ones. One example for sentiment analysis application can even be predicting how markets will shift due to financial news.

On one side there is interest to assign sentiment to the whole document and on the other, there is interest to assign sentiment to each entity mentioned in the document separately. We will focus on the more scarcely researched second option, called aspect based sentiment analysis, processing a dataset constructed of Slovene news article.

2 Related work

We reviewed the proposed literature and some additional papers, which research the problem of aspect-based sentiment analysis. Across all papers, the majority of recently popular approaches for natural language processing were used.

First, we read a paper (Asghar et al., 2014), which mostly focuses on data preprocessing and feature extraction. This paper along with most of the others we read recommends heavy preprocessing of the initial texts, including punctuation and stop word removal, transforming letters to lowercase, part of speech tagging and lemmatization. However (Bučar, 2017), which analyzes Slovene documents, recommends that we omit stop word and capitalization removal.

Most of the approaches take advantage of machine learning models, while (Sweeney and Padmanabhan, 2017) also tried predicting sentiment with lexicons. (Biyani et al., 2015) divided the classification problem into several binary classifications, where (Ding et al., 2018) used multiclass classification approach. Machine learning models used for classification ranged from Naïve Bayes (Bučar, 2017), Support Vector Machines (Tang et al., 2016), to neural networks (Jebbara, 2016) and transformers (Yang et al., 2019). There were also a few novel approaches. (Jebbara, 2016) created custom sentiment embeddings, (Wallaart and Frasincar, 2019) took the ontology approach, where they transformed words into aspects and classified sentiment based on domain specific rules. (Guha et al., 2015) used a special form of clustering instead of word embeddings. (Hercig et al., 2016) approached the problem with unsupervised methods. Intel's NLP architect, which is based on (Mamou et al., 2018), uses semi-supervised learning, where a domain specific opinion lexicon is automatically created and can be then manually corrected if needed.

Besides (Bučar, 2017) we also reviewed another lexicon approach on Slovene texts (Kadunc and Robnik-Šikonja, 2017). We also briefly went through (Bučar, 2017) doctoral dissertation, during which he collected the data set we are now analyzing.

3 Data description

The corpus we will use is the SentiCoref 1.0 (Žitnik, 2019) which consists of 837 documents and 433 thousand tokens which were selected from SentiNews 1.0 corpus (Bučar, 2017). The text contents of those selected documents are from five different Slovene news portals. The SentiCoref corpus is already annotated with following data for each token:

- location of token in text,
- type of named entities (person, organization or location),
- coreference to named entities,
- discrete sentiment range from very negative to very positive for each entity.

Most of the annotated sentiment in database is tagged as neutral. There is a lot of entities which span through a few tokens, and are therefore counted as multiple entities. We decided to count multi token entities as single entities.

The distribution for entities is following: 29 very negative, 1791 negative, 10726 neutral, 1701 positive and 24 very positive. There were also 301 entities, which did not have any reported sentiment, so we decided to exclude them from our dataset in future analysis.

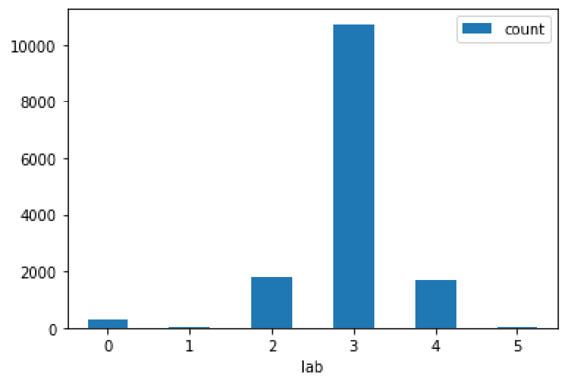

Figure 1: Sentiment distribution in SentiCoref 1.0 dataset

4 Methods

We started this project with limited knowledge regarding machine learning models used for advanced language processing. Throughout the duration of the project we gained some knowledge, but still stumbled upon obstacles which we weren't able to resolve.

Our initial idea was to get to the best options regarding preprocessing, feature extraction, modelling and evaluation through experimental testing. We intended to implement most of the perspective methods studied in related works and tried to develop a few of our own solutions in each of the steps mentioned above.

We first built a simple model, so that we got familiar with the data and algorithms used for natural language processing. We then tried to improve the before mentioned processes to get the best results possible.

We initially used less preprocessed text, as that appeared to yield better results in Slovene texts. We extracted the local context in several ways, first by using whole sentences where the entity appears, then taking into account only a fixed neighborhood of an entity. We then tried to get context by weighting the neighboring words based on their distance from the entity itself. In the end we used the Stanza dependency parser in hopes to get word relationship which would better represent entity context.

We used Slovene lexicons which under performed against a majority benchmark. We then applied simpler machine learning models such as Multinomial Naive Bayes, Logistic Regression and Support Vector Classifier. We also tackled the Neural Network and Transformer models, which unfortunately ended not very successfully.

Most of the previous papers used accuracy and F-score for evaluating their results. Initially we used accuracy as well but later changed it to weighted F-score, because of our concern about the distribution of our sentiment classes, where very polar classes represent a very small number of all cases.

We first had some discussions how to effectively extract entities with sentiment, because there were a few tokens with more than one belonging entity. There were also cases where a token was an entity but it had no sentiment, so we decided to skip such entities.

We first preprocessed the SentiCoref dataset, where we extracted entities from separate files. We then collected indices of tokens for each entity in each document, its type, as in person, location or organization. We then obtained the number of how many times the entity appears in the document and finally its sentiment. In the next stage we preprocessed the SentiNews document level dataset, where we extracted the text for each document, the length of the text, the news source, the sentiment average and standard deviation calculated from annotators and the news sentiment, which can only be negative, neutral or positive.

As already mentioned, we tried many approaches how to extract entity context. We concluded it would be best to use a dependency parser to extract relations among entities in the document. For that we used Stanza, which is capable of tokenization, part of speech tagging, lemmatization and dependency parsing. We first used all of the available options on document text and acquired the lemmas, POS tags and dependency heads for each token in a document. We then used the already collected entity tokens to gather first level contextual relations from the preprocessed documents. During this process we only allowed tokens with certain POS tags to be added to the context. This way we automatically excluded stop words or other irrelevant words.

In the next step we created the final dataframe which was then used for modelling. We joined the features from both the SentiCoref and SentiNews dataset and added some of our own features. We used one hot encoding to get binary features from categorical ones, such as entity type, news source and document sentiment. We believed that the context of entities would be useful, if it included positive or negative words. We used Slovenian lexicons(Kadunc and Robnik-Šikonja, 2017), to determine the polarity of words. We also thought that if a context word was an adjective or an adverb, the entity would more likely be polar. By looking at the sentiment distribution according to our features, if a word was negative, the distribution of sentiment would favor the negative sentiment. If there was an adverb included in context it was more likely than usual that the world was polar. Surprisingly adjectives and positive words didn't have much effect on sentiments.

We tried a few different ways to acquire context of entities, for which we believed it would be the most significant in correctly classifying entity sentiment. The entity neighborhood approach didn't yield good result because too many unrelated words appeared as a context. We gathered some important context with dependency parsing, but a lot of entities had no context when looking only at first level dependencies. We tried to expand this to two level dependency, but we stumbled upon a similar problem as with neighborhoods, too many unrelated words were categorized as context. In the end we just decided to stick with first level context, SentiNews and SentiCoref features.

4.1 Simpler models

In the initial stage of the project we only used the neighborhood of the words for modelling. First with lexicons and then with tf-idf vectorization of entity neighborhoods, we finished unsatisfied with the performance of our models. In the next stage we gathered additional entities and dependencies and used this features with models such as linear regression and multinomial naive Bayes. Unfortunately we arrived to similar performance issues, which will be discussed in the next section.

4.2 Neural networks

The first idea for figuring out sentiment for each token was to get token's context as plain text in sentences and somehow train them to neural network. For that reason we took each token's context from earlier methods and transformed it into sequence. The idea was to import Slovenian pretrained BERT model and fine tune it with our newly acquired embeddings. However during BERT model importing we encountered difficulties with locally installed Tensorflow version and so we decided to pursue other neural network models.

We then decided to continue training neural network models with fully connected neural network. As input feature vector we decided to use our final dataframe described in previous chapter. We had to modify some features in that dataframe. The reason for that was our design of fully connected neural network, which would only accept One Hot Encoded data instead of sparse data representation. Among data that we had to convert to One Hot Encoded data was also sentiment (result of our feature vector). We also decided to combine very negative and very positive sentiments with negative and positive sentiments respectively. With that step we hoped to achieve better results since we speculated predicting those classes with very low frequency would not be as good on neural networks.

After adjusting feature vector we ended up with feature vector of length 110 and 3 different classes of sentiment. For training fully connected neural network we chose Keras neural network library which ran on top of TensorFlow. Before neural network training we also had to split train and test dataset so we could correctly evaluate obtained model after training. Our fully connected neural consisted of one relu dense layer which connected to another softmax layer represents our discrete results (sentiment with three different classes). Since we used One Hot Encoded data our loss function was categorical crossentropy. We compared different batch sizes and numbers of epochs to come up with best results on described network.

We evaluated obtained model with weighted f1 score, similar to previous model evaluations.

5 Results and discussion

Results of our work are presented in table 1, and will be discussed separately in each of the two corresponding subsections.

Machine learning models	**Weighted F1**
Majority	0.650
Logistic regression	0.682
Linear SVC	0.676
Multinomial NB	0.687
FFN model (epochs)	**Weighted F1**
FFN (1)	0.720
FFN (3)	0.731
FFN (5)	0.725
FFN (7)	0.727
FFN (10)	0.735
FFN (15)	0.734
FFN (20)	0.724

Table 1: Different model result comparison

5.1 Simpler machine learning models

We used the same features for each model and arrived to similar results. Our results are better than the majority classifier, but in our opinion still subpar as a whole. There are still many things we could do to improve them. We should first check if our features actually bring any value to the model, and drop those who don't. We could then add additional features which provide us more information. One problem of our results is also the vastly different class distribution, which could be corrected with resampling. This problem can be seen in confusion matrix 2, where most of the classifications belong to neutral class, and very little to highly polar classes. We also didn't play much with model hyperparameters. Some models are also unable to see relations between features, and we could maybe achieve better results with models like neural networks.

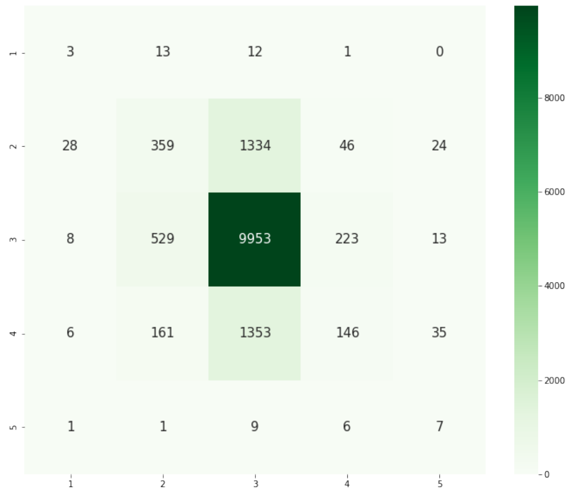

Figure 2: Confusion matrix for Multinomial Naive Bayes classification model

5.2 Neural networks

As mentioned we evaluated fully connected neural network with weighted F1 value. We observed that with increasing number of epochs (how many times train dataset is "trained" in network) F1 value also changed. If we would set too small or too big epochs value, results would generally be slightly worse. We speculated that with large value of epochs our model would overfit train data and evaluating with test data would therefore be worse. Weighted F1 values for different number of epochs are represented in table 1. Confusion matrix for described FFN model with number of epochs set to 10 is represented in figure 3.

We also recognised that using the same feature vector as we did for machine learning methods would bring similar problems as we described in previous chapter, namely very uneven class distribution. To counter that problem we could have payed more attention to selection of train database. With more similar class distribution in train database we could probably get better and more accurate weighted F1 results.

6 Conclusion

Overall we used and studied many different approaches to sentiment analysis. Since we started with very limited knowledge of machine learning models and neural networks for given problem, we

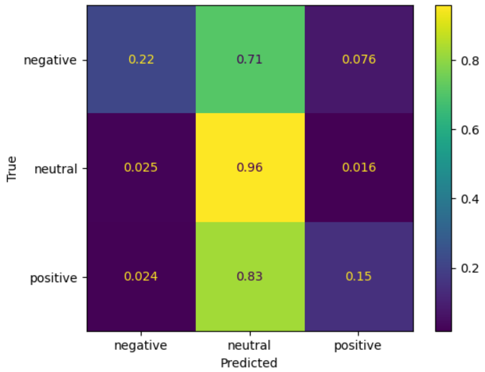

Figure 3: Confusion matrix for fully connected neural network model.

discovered and learned many new concepts in field of natural language processing.

We spent a lot of time on preprocessing and feature extractions which we discovered are very important for subsequent steps. However we acknowledge that there is still a lot of room for improvement there.

We tried a few different machine learning and neural network models. We regret not finishing BERT embedding model (with maybe better context) but due to lack of time and experience we had to continue with simpler neural network models. Similarly we also planned to implement word2vec model which is now left as a possible improvement of our methods.

References

Dr. Muhammad Asghar, Aurangzeb Khan, Shakeel Ahmad, and Fazal Kundi. 2014. A review of feature extraction in sentiment analysis. *Journal of Basic and Applied Research International*, 4:181–186.

Prakhar Biyani, Cornelia Caragea, and Narayan Bhamidipati. 2015. Entity-specific sentiment classification of yahoo news comments.

Jože Bučar. 2017. Manually sentiment annotated slovenian news corpus SentiNews 1.0. Slovenian language resource repository CLARIN.SI.

Jože Bučar. 2017. *Sentiment based classification of the web texts*. PhD dissertation, Fakulteta za informacijske študije v Novem mestu.

Jin Ding, Hailong Sun, Xu Wang, and Xudong Liu. 2018. Entity-level sentiment analysis of issue comments. pages 7–13.

Satarupa Guha, Aditya Joshi, and Vasudeva Varma. 2015. SIEL: Aspect based sentiment analysis in reviews. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 759–766, Denver, Colorado. Association for Computational Linguistics.

Tomáš Hercig, Tomáš Brychcín, Lukáš Svoboda, Michal Konkol, and Josef Steinberger. 2016. Unsupervised methods to improve aspect-based sentiment analysis in czech. *Computacion y Sistemas*, 20:365–375.

Soufian Jebbara. 2016. Aspect-based sentiment analysis using a two-step neural network architecture. volume 641, pages 153–167.

Klemen Kadunc and Marko Robnik-Šikonja. 2017. Slovene sentiment lexicon KSS 1.1. Slovenian language resource repository CLARIN.SI.

Jonathan Mamou, Oren Pereg, Moshe Wasserblat, Alon Eirew, Yael Green, Shira Guskin, Peter Izsak, and Daniel Korat. 2018. Term set expansion based NLP architect by intel AI lab. *CoRR*, abs/1808.08953.

Colm Sweeney and Deepak Padmanabhan. 2017. Multi-entity sentiment analysis using entity-level feature extraction and word embeddings approach. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 733–740, Varna, Bulgaria. INCOMA Ltd.

D. Tang, F. Wei, B. Qin, N. Yang, T. Liu, and M. Zhou. 2016. Sentiment embeddings with applications to sentiment analysis. *IEEE Transactions on Knowledge and Data Engineering*, 28(2):496–509.

Olaf Wallaart and Flavius Frasincar. 2019. *A Hybrid Approach for Aspect-Based Sentiment Analysis Using a Lexicalized Domain Ontology and Attentional Neural Models*.

Heng Yang, Biqing Zeng, JianHao Yang, Youwei Song, and Ruyang Xu. 2019. A multi-task learning model for chinese-oriented aspect polarity classification and aspect term extraction.

Slavko Žitnik. 2019. Slovene corpus for aspect-based sentiment analysis - SentiCoref 1.0. Slovenian language resource repository CLARIN.SI.