# Aspect-based sentiment analysis

**Miha Bizjak, Anže Gregorc, Rok Grmek**
University of Ljubljana
Faculty for computer and information science
Večna pot 113, SI-1000 Ljubljana
mb9232@student.uni-lj.si, ag9497@student.uni-lj.si, rg6954@student.uni-lj.si

## Abstract

TODO

## 1 Introduction

Online news, forums and social media are a place for everyone to read and write articles and posts across various domains. People can also leave comments and giving their opinion and express their feelings about the topics. That leads to a huge amount of text content. That is probably why natural language analysis is currently a hot topic around the world. We wanted to extract useful information out of large amount of text data.Since we have no time for reading all the words that are written nowadays, we hope to build a good computer program to do that for us. In this project we chose to do aspect-based sentiment analysis. Our task is to get the subjective information from text material that refer to a entity with the use of natural language processing and other methods. An entity is considered as a person, organization or a location and can be represented multiple times in one document or a sentence and there could be more entities in one document.

For the given task, we decided to test multiple approaches and develop different models for predicting the sentiment for each entity. We will first define some really simple models as a starting point, and then we will try to derive some more complex models. All of them will be targeting the Slovene language, and we will evaluate each of the models on the Slovene corpus for aspect-based sentiment analysis - SentiCoref 1.0 (Žitnik, 2019).

## 2 Related work

The main challenge of entity-based analysis is how to find words that describe the entity and identify if contributes to positive or negative sentiment to a given entity. A lot of related work tried to predict sentiment of the whole document. But in many cases, a text can describe the polarity of more entities. That is why we suggest that sentiment analysis is done on entity level. Since our task is more specific we focused more on methods that identify entities.

In the paper (Ding et al., 2018) they developed an entity-based sentiment analysis SentiSW and tested it on issue comments from GitHub. SentiSW can classify issue comments into <*sentiment, entity*> tuples. They evaluate the entity recognition by manually annotation and it achieves 75% accuracy. The main pipeline of this tool is preprocess (words removal, words replacing, stem), feature vectorize (TF-IDF, Doc2vec), classifier (random forest, bagging and other supervised machine learning methods) and entity recognition (rule-based method).

The use of word embeddings provide powerful methods for semantic understanding without the need of creating large amounts of annotated test data. The paper (Sweeney and Padmanabhan, 2017) enhanced the word embeddings approach with the deployment of a sentiment lexicon-based technique to appoint a total score that indicates the polarity of opinion in relation to a particular entity. They associate a given entity with the words describing it and extracting the associated sentiment to try to infer if the text is positive or negative in relation to the entity.

As stated in the paper (Song et al., 2019), a lot of the existing approaches are modelling context and target words with RNNs and attention. This paper addresses the issues with RNNs and proposes an Attentional Encoder Network for modeling the semantic interactions between target and context words. The paper also addresses the label unreliability issue. The proposed model with the use of pre-trained BERT embedding achieved state-of-the-art results while still being a relatively

lightweight model.

Another successful approach that utilizes the BERT model is described in the paper (Sun et al., 2019). In this paper, the authors tried a few methods, where they were generating an auxiliary sentence for each prediction. They basically converted the aspect-based sentiment analysis problem into a sentence-pair classification task.

## 3 Methods

### 3.1 Dataset

The SentiCoref 1.0 dataset contains 837 documents with annotations of named entities (31,419 entities in total) and sentiment annotations for each entity. For each entity, a sentiment value from 1 to 5 is assigned. The distribution of sentiment labels in the dataset is shown in Table 1.

| Sentiment label | Entity count |
|---|---|
| 1 - Very negative | 30 |
| 2 - Negative | 1801 |
| 3 - Neutral | 10869 |
| 4 - Positive | 1705 |
| 5 - Very positive | 24 |

Table 1: Entity sentiment distribution in the SentiCoref 1.0 dataset.

### 3.2 Models

All developed models are described in this section. First two models are really simple and have no practical use - we implemented them as a starting point in order to test the entire evaluation framework, and to have a reference performance for comparing other more complex models.

#### 3.2.1 Random model

The Random model simply assigns random sentiment (1 - 5) to each entity, without knowing anything about the actual sentiment. Assigned sentiments are uniformly distributed, and even the distribution of the sentiments in the dataset was not taken into account. We are expecting that this model would serve as a reference for the lowest possible evaluation performance.

#### 3.2.2 Majority model

The Majority model assigns neutral sentiment (3) to all entities. The idea for this model came from a fact that around 75% of the entities in the dataset are labeled with the neutral sentiment. Therefore, we are expecting decent performance because of the distribution of the sentiments in the dataset. We will also use this performance as a reference for comparing other models. More complex models are expected to extract more knowledge from the given data, and they should be performing better than this one.

#### 3.2.3 Lexicon Features model

TODO

- The model uses a random forest classifier with the following features:
  - number of positive words up to 5 words before/after each entity occurrence,
  - number of negative words up to 5 words before/after each entity occurrence,
  - number of positive words for which the current entity is the closest,
  - number of negative words for which the current entity is the closest,
  - number of positive words in sentence,
  - number of negative words in sentence,
  - number of positive words in the text,
  - number of negative words in the text,
  - number of different entities in the text,
  - number of occurrences of the entity.
- The optimal parameters for the model are chosen using grid-search with 5-fold cross-validation.
- Sentence splitting is done using the pre-trained Punkt sentence tokenizer (Kiss and Strunk, 2006) for Slovene provided with the `nltk` Python package (Bird et al., 2009).

#### 3.2.4 BERT model

The well known BERT (Bidirectional Encoder Representations from Transformers) model (Devlin et al., 2018) can be used for variety of natural language processing tasks, including the sentiment analysis. We decided to utilize the pre-trained multilingual BERT embedding layers and combine them with additional convolutional layers and some dense layers at the end for direct classification of the sentiment.

In order to prevent to much variation in the length of inputs (the number of mentions can vary significantly from entity to entity), we presented the idea of treating each mention of an entity as a separate learning case. Therefore we constructed

a training set such that each input was represented with a set of words from the neighbourhood of the entity mention, and each output was a sentiment class (1 - 5) of the mentioned entity. In the prediction stage, where the goal is to predict the sentiment for the entity and not for all individual mentions of the entity, we decided to sum predictions over all mentions of the entity, and we select the class with maximum support. The main assumption here was, that all mentions of an entity has similar (hopefully equal) sentiment. Therefore, we may expect some issues if an entity would have extremely negative sentiment in some mentions and extremely positive sentiment in other mentions, but the average sentiment for this entity would be labeled neutral in our dataset. In this case, we would also assign neutral sentiment to all mentions of this entity when constructing the training samples, although some sentiments could be extremely negative or extremely positive.

The architecture of our model is shown in Figure 1. The model is roughly constructed from three parts - first part converts a sequence of words into word embeddings, where the pretrained BERT is used; the second part is convolving with different filters over the sequence of embeddings; and the third part is used for classifying the features, that were extracted with the convolution, into one of the five sentiment classes.

The sequence of words is first extracted from the neighbourhood of the entity mention (parameter *n_words_left_right* defines how many words to the left and right are used). Then before using the BERT embedding layers, we encode the sequence of words and pad the sequence to a fixed length of 128. After obtaining the BERT embeddings, we use three parallel convolutional layers with *conv_filters* number of filters, where the kernels are of sizes 1, 2 and 3 correspondingly. To each filter we apply the max pooling, and then we concatenate the results into a vector of size $3 \cdot$*conv_filters*. The concatenated vector is then put through the dense layer with *dense_units* number of neurons, then we apply the dropout to prevent overfiting, and lastly we use the dense layer with 5 neurons and the softmax activation for the final classification (other neurons are using the relu activation).

The model fitting procedure was done with different dropout rates (parameter *dropout_rate*) in batches of size *batch_size*, and with *epochs* number of training repetitions. During the fitting, the weights from BERT embedding layers were not updated.

#### 3.2.5 Dependency model

TODO

### 3.3 Evaluation

TODO

- 2/3 train, 1/3 test split
- Implemented measures: Accuracy, Precision, Recall, F1 score.

## 4 Results

TODO

### 4.1 Lexicon Features model

TODO

### 4.2 BERT model

The model was developed with variety of free parameters. We tested different configurations in order to see the impact of each parameter on the model performance. For the starting point we set the parameters as follows: *n_words_left_right*: 6, *conv_filters*: 100, *dense_units*: 256, *dropout_rate*: 0.2, *batch_size*: 128, *epochs*: 5. Then we evaluated several experiments were we were changing one parameter at a time and left all others fixed. All the results are given in Tables 2, 3, 4, 5, 6, 7 and in Figure 2.

| | F1 score | Precision | Recall | RMSE |
|---|---|---|---|---|
| 1 | 0.6459 | 0.5844 | 0.7503 | 0.5072 |
| 2 | 0.6541 | 0.6131 | 0.7495 | 0.5124 |
| 3 | 0.6470 | 0.6103 | 0.7474 | 0.5126 |
| 4 | 0.6628 | 0.6348 | 0.7250 | 0.5544 |
| 5 | 0.6631 | 0.6383 | 0.7233 | 0.5597 |
| 6 | 0.6734 | 0.6519 | 0.7114 | 0.5812 |
| 7 | 0.6692 | 0.6465 | 0.7083 | 0.5775 |

Table 2: Evaluation results of the BERT model with respect to parameter *n_words_left_right*.

| | F1 score | Precision | Recall | RMSE |
|---|---|---|---|---|
| 50 | 0.6596 | 0.6455 | 0.7330 | 0.5527 |
| 100 | 0.6734 | 0.6519 | 0.7114 | 0.5812 |
| 150 | 0.6625 | 0.6364 | 0.7068 | 0.5835 |

Table 3: Evaluation results of the BERT model with respect to parameter *conv_filters*.

| | F1 score | Precision | Recall | RMSE |
|---|---|---|---|---|
| 64 | 0.6461 | 0.5661 | 0.7524 | 0.5058 |
| 128 | 0.6585 | 0.6305 | 0.7262 | 0.5527 |
| 256 | 0.6734 | 0.6519 | 0.7114 | 0.5812 |

Table 4: Evaluation results of the BERT model with respect to parameter *dense_units*.

| | F1 score | Precision | Recall | RMSE |
|---|---|---|---|---|
| 0.1 | 0.6745 | 0.6510 | 0.7131 | 0.5815 |
| 0.2 | 0.6734 | 0.6519 | 0.7114 | 0.5812 |
| 0.3 | 0.6686 | 0.6525 | 0.6905 | 0.6077 |

Table 5: Evaluation results of the BERT model with respect to parameter *dropout_rate*.

TODO (explain results)

### 4.3 Dependency model

TODO

## 5 Discussion

TODO

## 6 References

## References


Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jin Ding, Hailong Sun, Xu Wang, and Xudong Liu. 2018. Entity-level sentiment analysis of issue comments. In *Proceedings of the 3rd International Workshop on Emotion Awareness in Software Engineering*, pages 7–13.

Tibor Kiss and Jan Strunk. 2006. Unsupervised multilingual sentence boundary detection. *Computational linguistics*, 32(4):485–525.

Youwei Song, Jiahai Wang, Tao Jiang, Zhiyue Liu, and Yanghui Rao. 2019. Attentional encoder network for targeted sentiment classification.

Chi Sun, Luyao Huang, and Xipeng Qiu. 2019. Utilizing BERT for aspect-based sentiment analysis via constructing auxiliary sentence. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 380–385, Minneapolis, Minnesota. Association for Computational Linguistics.

Colm Sweeney and Deepak Padmanabhan. 2017. Multi-entity sentiment analysis using entity-level feature extraction and word embeddings approach. In *RANLP*, pages 733–740.

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


| | F1 score | Precision | Recall | RMSE |
|---|---|---|---|---|
| 32 | 0.6533 | 0.6237 | 0.7410 | 0.5260 |
| 64 | 0.6592 | 0.6380 | 0.7334 | 0.5440 |
| 128 | 0.6734 | 0.6519 | 0.7114 | 0.5812 |

Table 6: Evaluation results of the BERT model with respect to parameter *batch_size*.

| | F1 score | Precision | Recall | RMSE |
|---|---|---|---|---|
| 5 | 0.6734 | 0.6519 | 0.7114 | 0.5812 |
| 10 | 0.6750 | 0.6570 | 0.7015 | 0.6005 |
| 15 | 0.6636 | 0.6548 | 0.6743 | 0.6430 |

Table 7: Evaluation results of the BERT model with respect to parameter *epochs*.

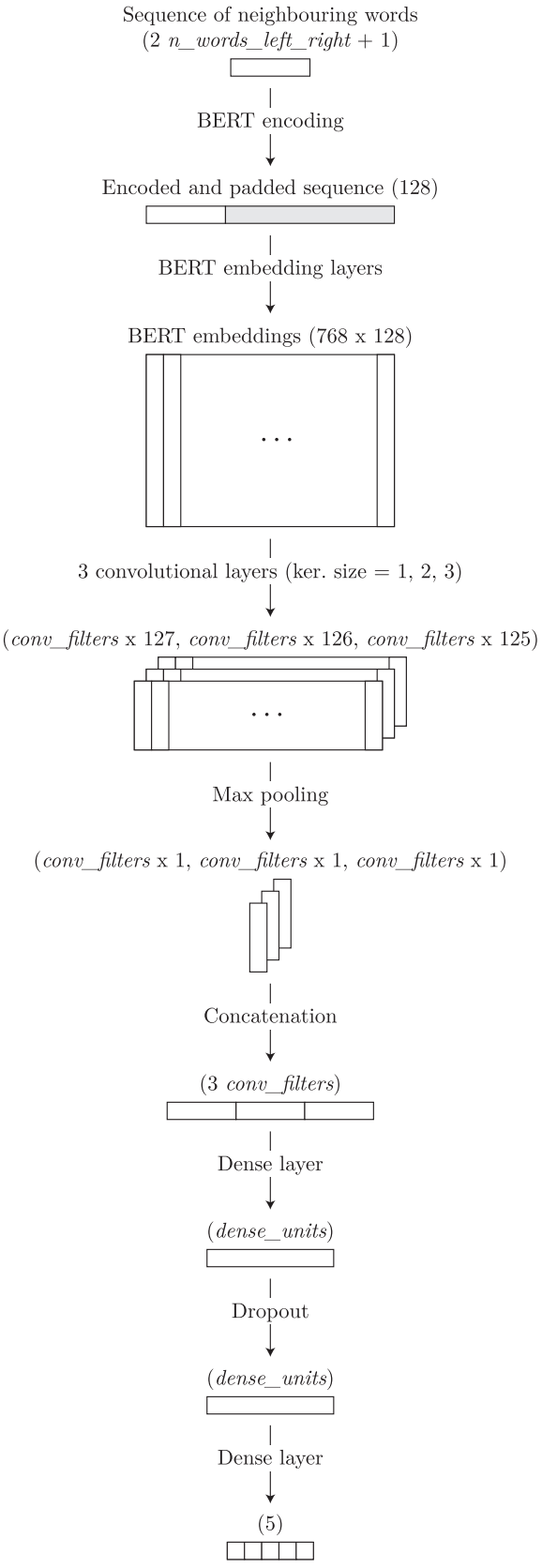


Figure 1: Architecture of the BERT model.

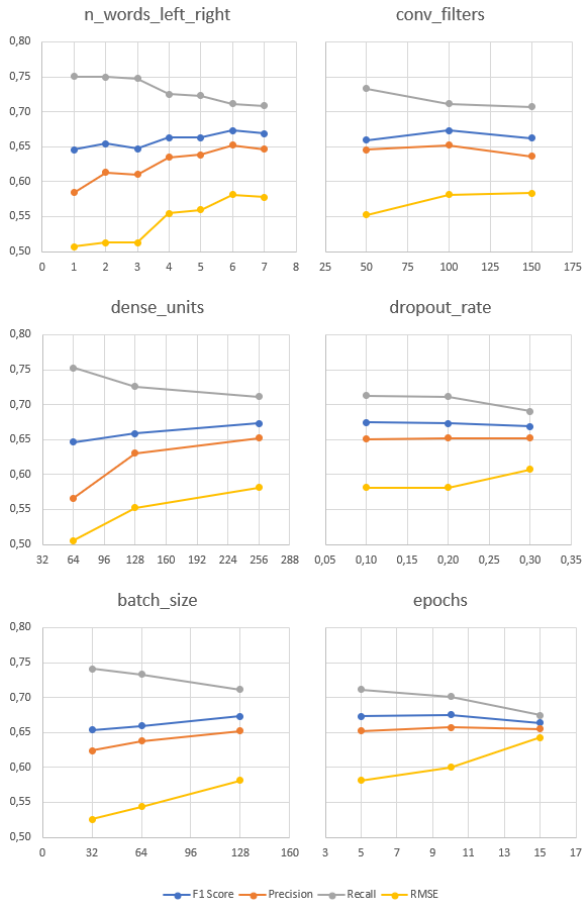


Figure 2: Evaluation results of the BERT model.