Natural Language Understanding Project Report
Sentiment analysis with BERT

Antonella Rech 232248

University of Trento

`antonella.rech@studenti.unitn.it`

Abstract

This document represents the report for the final project of the Natural Language Understanding course enrolled at the University of Trento. The main goal was to implement a sentiment analysis model that consisted of two specific tasks: subjectivity detection and polarity classification.

1. Introduction

Sentiment analysis is an automated approach to natural language processing that aims to analyse the text and identify the emotional tone of the writer. A common method of categorization consists in determining if the nature of the sentiment is positive, negative, or neutral. Accordingly, to approach this task, two sub-tasks are defined: a subjectivity detection phase and a polarity classification step. The first one is addressed as a pre-processing step that aims to filter the neutral elements of the text. The second one, instead, is solved in a classical way by implementing a classifier to determine if the feeling is positive or negative. In this particular project, both sub-tasks are implemented using classical machine learning and deep learning methods. For the machine learning approach, sentiment analysis is commonly performed using algorithms such as Naive Bayes and Support Vector Machine. A text classifier is trained using vectors, which convert the text into trainable features for the algorithm. However, one big disadvantage of this approach lies in the necessity of finding relevant information in the text to use such models. Often this phase is extensive and time-consuming and involves different cleaning steps, such as removing punctuation, stopwords, etc. The recent introduction of BERT, a transformer-based algorithm developed by Google in 2018, has allowed a better representation of the words in the text using also information from the context through the use of encoder which remove the necessity of pre-processing. Then, for the deep learning approach, this algorithm is performed for both subjectivity detection and polarity classification.

2. Task Formalisation

The main goal of the project is to resolve a sentiment analysis problem composed of a first part called subjectivity detection and a second part called polarity classification. The datasets assigned for this task are the movie review and the subjectivity datasets that can be easily accessed through the NLTK package. The implementation of the models was left to be chosen by the students. Then a baseline was developed similarly to the approach proposed *In Proceedings of the ACL* [2]; meanwhile, a fine-tuned BERT was chosen as a comparison. In the next sub-sections will be explained the two steps of the task.

3. Datasets

In this section are provided the two datasets used. Both of them were introduced in Pang/Lee ACL 2004 paper [2] and can be downloaded using the NLTK Downloader (site: `https://www.nltk.org`).

3.1. Subjectivity detection dataset

The subjectivity dataset is a collection of 5000 subjective and 5000 objective processed sentences. Specifically, it contains 2 files that collect quotes and short plots about movies, which respectively represent the subjective and objective data. Each line in these two files corresponds to down-cased single sentence. The entire dataset is used to train a classifier in order to perform the subjectivity detection task at sentence-level on the movie review dataset. The version used is the subjectivity dataset v1.0.

3.2. Polarity classification dataset

Movie review is a collection of labeled movie reviews. It contains 2 folders that subdivide the reviews into two categories defined as neg and pos. In particular, 1000 negative reviews can be found in first folder and 1000 positive reviews in the second one. Each review is a document in format .txt file. The dataset is split into 1600 reviews for the training and 400 for the testing. Moreover each set contain the same amount of positive and negative reviews to maintain a constant distribution of samples and permit the classifier to learn equally for both classes. In the machine learning approach also a cross-validation is performed with a number of 5 split to keep the proportion between train and test set expressed above. The version used is the polarity dataset v2.0.

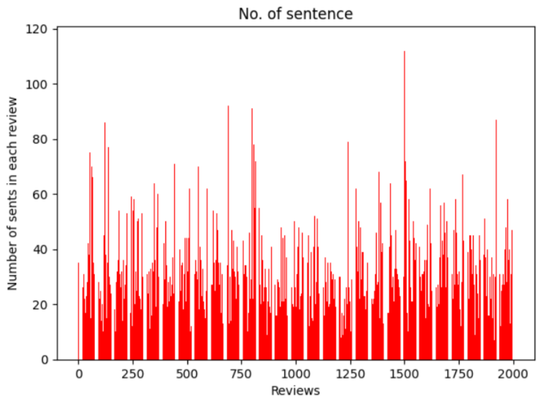
3.3. Statistics

In the table below (Table 1) are summarized the specific of each dataset.

Table 1: *Datasets summary statistics*

dataset	vocab len	no.sample	labels
subjectivity dataset	23906	10000	subj,obj
polarity dataset	39768	2000	pos,neg

The following graph shows the distribution of sentences for each movie review. This will be useful to observe the reduction of the number of sentence after performing the subjectivity detection.



4. Models

The following workflow is the general approach used in both models:

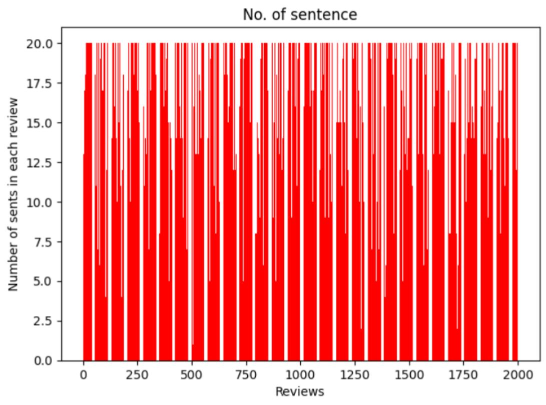
- *A Preprocessing step, if needed*
- *Implementation of an algorithm for subjectivity detection trained on the whole subjectivity dataset*
- *Performing the detection on the movie review dataset by removing the objective sentences using a certain criterion*
- *Implementation of an algorithm for the polarity detection trained and tested on the movie review dataset*

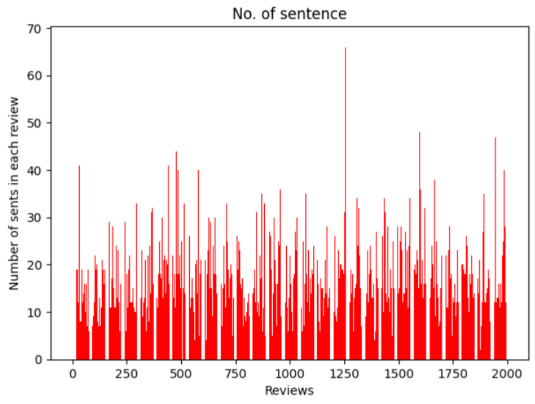
4.1. Machine learning approach

The model presented is composed of a first phase of preprocessing that consists of removing stopwords and punctuation, pos-tagging, and obtaining stem words through lemmatization. This will permit the creation of a clean and consistent dataset that is more manageable for the feature extractor. In addition, a way to handle the negations is tested by assigning a negation mark only on the next following word; this method is similar to the one expressed in *An Improved Text Sentiment Classification Model Using TF-IDF and Next Word Negation* [3]. The features are extracted using the CountVectorizer with min set to 1 to simulate a simple bag of words and the TfidfVectorizer with the idf set to True for a more weighted representation. Both extractors are also tested using different ranges of ngrams, specifically (1,1), (1,2), and (1,3). Then the best results are found considering unigrams, bigrams and trigrams. For the subjectivity detection and the polarity classification, three different algorithms are chosen: Naive Bayes, SVM, and logistic regression. Based on time and accuracy, the best performance received is from Naive Bayes.

4.1.1. N-best sentences criterion

The criterion used is similar to the one expressed here [2]. Two different ways to extract the N-best sentences are proposed. One method extracts a fixed number of the best subjective sentences from each review. To assign the score, a function to predict the class probability is used. In particular, the N tested is equivalent to 20. The second method, instead, consider the different length of each review and take a percentage of N-best sentences, specifically was found a good performance assigning N to 60%. Additionally, another control on the score is taken by filtering out all the sentences assigned to the subjective class with a probability less than 0.6. By applying one of the two criteria expressed, no substantial difference was found in the evaluation of the prediction. In the figures below, we can see the results of the two methods.





4.2. Deep learning approach

The Bidirectional Encoder Representations from Transformers, also called BERT, represent a machine learning model used for NLP tasks. One distinguishing characteristic of this model is the ability to learn information from both the left and right sides of a token's context. Applying this bidirectional training can give a deeper sense of language context. Unlike other models that use the same method, Bert makes use of Transformer, an attention mechanism that learns contextual relations between words in the text. A transformer embeds lists of tokens into vectors, and then they are processed inside the network. For this project, I used the base model which is defined with 12-layer, 768-hidden, 12-heads, 110M parameter. In particular, I fine-tuned a pre-trained version of BERT called BertForSequenceClassification first for the subjectivity detection and then for the polarity classification. The tokenization process require some important steps to make the data compatible with the network, the BERT encoder were then used:

- Add special tokens at the start and the end of the sentence ([CLS] + sentence + [SEP])
- Pad and truncate a sentence to a fixed given length (in my case max_len = 64)
- Create an attention mask to distinguish between real token and pad

Finally, the encoder produces a vector of embeddings with dimension [batch × length × hidden size] , with hidden size 768.

4.2.1. *Subjectivity detection criterion*

The criterion used here is a method to decide which sentences are to be kept. The sentences are defined as subjective if their probability score is beyond a specific threshold. In this case is set to 0.60 to remove also the weakest sentences that were score as subjective. To define the probability score, a sigmoid is performed at the logit level of the network where the logits are the outputs of the network of dimension [batch × length].

4.2.2. *Polarity classification criterion*

One review is assigned a positive label if the sum of all the predicted positive sentences in it is more than half the total predicted sentences in it.

4.2.3. *Parameters*

In the following list are reported all the parameters tuned for the training:

- learning rate = 1e-4, epsilon 1e-8
- batch size = 32
- subjectivity detection: no.epochs 4
- polarity classification: no.epochs 6

The implementation of a linear scheduler was tested to dynamically change the learning rate during the training steps and see if it would improve performance.

4.2.4. *BERT truncated*

BERT takes into account only sentences with a maximum length of 512 tokens. To overcome this limitation, I implemented another method for polarity classification that takes into account not just single sentences but the entire document. The method chosen consists of truncating the document into a single sentence of length 512, as mentioned in *How to Fine-Tune BERT for Text Classification?* [5]. The authors of the paper empirically found that most of the time the salient parts of a document are situated at the start and at the end of it; if we truncate the reviews, we will be able to perform well even with fewer tokens. In this work, 50 words are chosen for the head and 462 for the tail giving more weight to the last sentences since subjective phrases are found at the end after removing the objective sentences from the dataset.

5. Evaluation

The main metric used in this project to evaluate the performance of the baseline model and the BERT model is the accuracy. The accuracy score is calculated by dividing the number of correct predictions by the total prediction number; in the table below are reported the results all achieved through Google Colab.

Table 2: *Results*

Detector	time	Classifier	time	accuracy
Naive Bayes	1.5s	Naive Bayes	4.8s	86.6%
Naive Bayes	1.5s	SVM	241s	87%
Naive Bayes	1.5s	LogisticReg	24.5s	85.1%
BERT	87.6s	BERT	27s	91.25%
BERT	87.6s	BERT-trunc	15s	91%

For the baseline reported in Table 2, there are two other aspects to consider: the ngram range and the feature extractor. Using the TfidfVectorizer, with the inverse-frequency enable, instead of a CountVectorizer improved the performance of the Naive Bayes from 83.8% to 85% of accuracy. Additionally, considering a range of unigrams, bigrams, and trigrams instead of only unigrams, the accuracy was able to reach 86.6%. Finally, as shown in the Table 2 the best accuracy between the variants of the baseline is given by the SVM model, but the fastest algorithm was instead the Naive Bayes. For the same reason, the detector was trained only on the Naive Bayes, which gave approximately the same results as the other models. Finally, the accuracy reported were calculated using cross-validation to obtain the averaging results.

As we can see, the BERT model was able to reach a higher level of accuracy compared to the baselines. The results vary by about 1 or 2 percent depending on how the dataset is subdivided, and the time indicated on the table represents how long the evaluation step took. By adding the linear learning rate scheduler, the accuracy reached from the model was higher by 1% with an average training loss of 0.22. One main advantage of BERT is that it is already pretrained on a large amount of data (trained on BooksCorpus: 800M words), and even if it is applied on a small dataset can still have a good performance. Another good aspect is the embedding process that take into account also the surronding which give a deeper representation to the words respect to a feature extraction based on their frequency. However, the principal limitation of the proposed work comes from the incapacity of BERT to take input sequences with more than 512 tokens; in fact, the classification of the reviews was based on a sentence-level prediction for each sentence contained in it without taking into consideration any correlation between them. The power of context was then used only on the sentence level, instead a context from a document-level approach will be surely enhance the performance of the model. Consequentially, Bert-trunc presents a variant of Bert that tries to overcome this problem by truncating the reviews into sentences of length 512 and training the classifier on them. One achievement of this test was to obtain the same accuracy results as BERT while using less information due to the truncation. Although this method risks missing some important information if the text is structured differently and missing some correlations between the sentences by giving a non-complete context representation.

6. Conclusion

This work presented a comparison of the BERT-based model with the improved baseline on the movie review and subjectivity datasets. Results showed that the fine-tuned BERT can still obtain better results than a standard machine learning approach with a preprocessing phase. However, It could be interesting to try a different pipeline for the Bert-based model that can extract more specific information not only on the sentence-level but also on the document-level; in this way, the sentence will be weighted based on the context present in the document.

7. References

[1] NLTK Documentation, `https://www.nltk.org/_modules/nltk/sentiment/util.html`

[2] Bo Pang, Lillian Lee, *A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts*

[3] Bijoyan Das, Sarit Chakraborty, *An Improved Text Sentiment Classification Model Using TF-IDF and Next Word Negation.*

[4] Transformers 3.3.0 documentation, `https://huggingface.co/transformers/v3.3.1/training.html`

[5] Chi Sun, Xipeng Qiu, Yige Xu, Xuanjing Huang, *How to Fine-Tune BERT for Text Classification?*.