Natural Language Processing on Movie Reviews

Vičević Dominik and Mikulić Mateo

***Abstract*— In this paper, we investigate how NLP (Natural Language Processing) sentiment analysis can be applied to movie reviews. We provide an overview of possible approaches to sentiment analysis using machine learning methods, from simple to more complex architectures. For simple architectures, we used Naive Bayes, Maximum Entropy, and Support Vector Machines algorithms. As an improvement to classical methods, we applied Recurrent Neural Networks (RNNs) in the form of SimpleRNN, LSTM, and GRU. The use of RNNs was introduced to provide context to machine learning methods. To improve on RNNs, we used NLP's state-of-the-art Transformer architectures, BERT and DistilBERT. As expected, Transformer architectures had the best test accuracy. DistilBERT was the best with 93.5%. Classical methods surpassed our expectations with accuracies of 85.3%, 87.8%, 85.5% for NB, ME, and SVM, respectively.**

I. INTRODUCTION

It has become essential to comprehend the sentiments represented in textual data in a number of fields, including marketing, customer feedback analysis, and public opinion tracking. Particularly, movie reviews provide insightful criticism for both spectators and creators. The use of Natural Language Processing (NLP) methods, particularly sentiment analysis, has become a potent tool for revealing the emotions concealed in movie reviews.

In this paper, we investigate how NLP sentiment analysis can be applied to movie reviews. Our research strives to elucidate methods, problems, and developments in sentiment analysis.

The paper provides an overview of sentiment analysis applicability to movie reviews, difficulties unique to this field, and approaches used. The efficiency of sentiment analysis methods is confirmed by a case study that makes use of a sizable corpus of movie reviews.

II. METHODOLOGY

As a starting point, we used the reference "Thumbs up? Sentiment Classification using Machine Learning Techniques" [1]. On a revised version of the same dataset [2] (referred to as dataset A), we reimplemented the methodologies from the original study. We also intended to test the effectiveness of similar techniques on a dataset with significantly larger dimensions [3] (referred to as dataset B). Ultimately, we wanted to test how state-of-art techniques would compare to baseline results from a 2002 study while also attempting to improve upon them.

*This work was done for Evolutional Computing Graduate Course on Faculty of Engineering in Rijeka

A. *Classical methods*

Because of our starting point, we decided to reimplement methods used in the paper [1]. These methods are:

- **Naive Bayes**
- **Max Entropy**
- **Suppported Vector Machines**

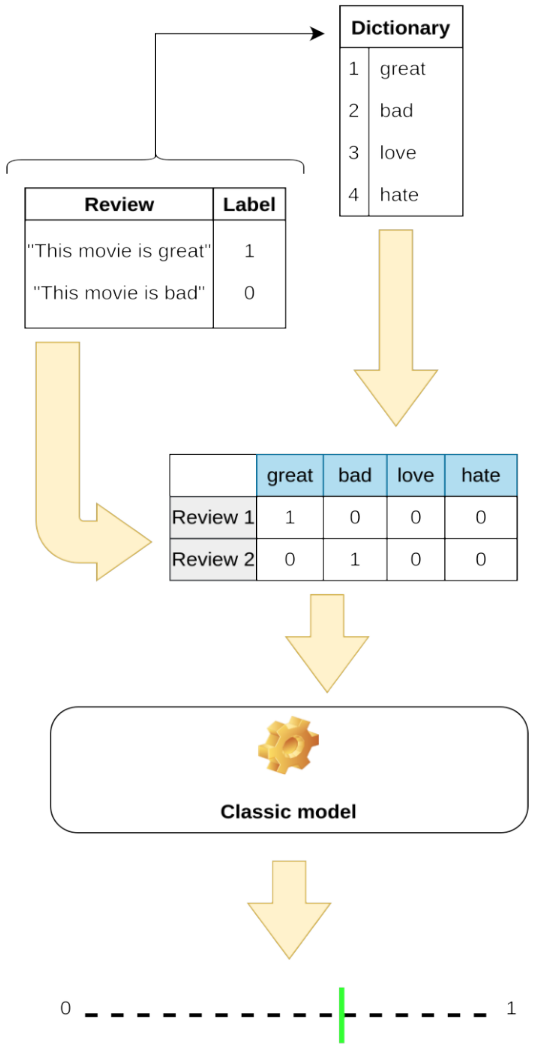

Fig. II.1. Classic models pipeline

Figure II.1 shows the pipeline used for implementing classical methods. First we created a dictionary that contains all the words we considered for sentiment analysis. We did this by filtering words for each review, then, we added the words that appeared more than 5 times to the dictionary.

We filtered the words using stop words. Stop words are commonly used words that can be ignored (examples of stop words include ”the,” ”a,” ”an,” etc). We then filtered the reviews to only contain the words that are in the dictionary.

Using the filtered reviews we create a sparse matrix where the columns are all the words in the dictionary, and the rows are the reviews, therefore a cell in the matrix can contain the number 1 if the word from the dictionary is present in the review, or 0 if it is not. It is important that a sparse matrix is used; because the matrix created is filled with mostly zeroes, naively storing all that data would cost us a lot of memory while using dataset B, which has a dictionary of 50,128 words and 50,000 reviews. A sparse matrix alleviates this memory cost by storing only the non zero elements of the matrix and their position. The created sparse matrix is then fed into the models for training.

The problem with the mentioned classical methods is that they do not consider the context the words are in, they only learn if the word in question is used more in positive or negative sentiment reviews and make their prediction based on that. To add context into account we need to look to more modern and complex methods.

B. *Modern methods (Recurrent Neural Networks)*

As a more modern approach, we used Recurrent Neural Networks (RNN). When our baseline paper [1] discussed incorporating more sophisticated techniques for modeling context into sentiment classification, the decision to employ RNNs was taken. When making predictions, RNNs take the entire sequence into account rather than just one attribute.

The pipeline used to implement RNNs is shown on figure II.2. The dictionary creation was identical to the classical methods, so we won’t explain it again here. As input to the model, words in the reviews were represented as indices of words in the dictionary. Reviews were zero-padded to the desired length. Here we tried two approaches: removing outliers or taking the length of the longest review in the dataset. Taking the length of the longest review is trivial so it doesn’t need an explanation. Removing outliers was achieved using the IQR (interquartile range). First, we need to calculate the IQR, which is obtained by subtracting the 25th percentile from the 75th percentile. Then, we discard outliers that are less than the lower threshold

$$25th\_percentile - 1.5 * IQR \tag{1}$$

and greater than the upper threshold

$$75th\_percentile + 1.5 * IQR \tag{2}$$

We then took the length of the longest review within the threshold as the input size.

The dimensionality of the input vectors was reduced by using an embedding layer. The idea behind embeddings is that similar data would have comparable latent representations. These embeddings made by the embedding layer are sent to the version of RNN model used. RNN model is regularized with Dropout and L2 regularization. As the output layer we used a fully connected neuron with sigmoid activation function.

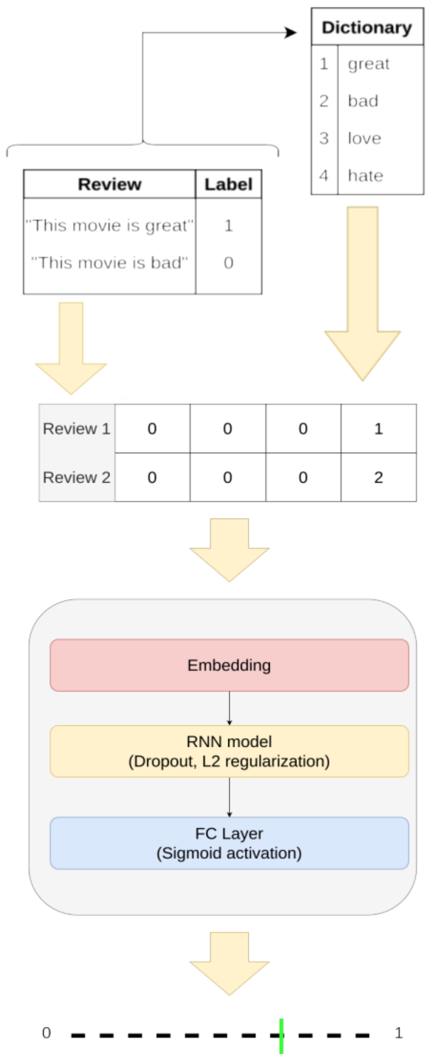

Fig. II.2. RNN models pipeline

- **SimpleRNN**

  We employed SimpleRNN as the simplest form of RNN.

- **LSTM (Long Short Term Memory)**

  LSTM was employed as more complex form of RNN because SimpleRNN behaved unpredictably which is expected by its short-term memory architecture. LSTMs in addition to short-term memory have capability of storing knowledge from prior inputs (long-term memory).

- **GRU (Gated Recurrent Unit)**

  As LSTMs were behaving somewhat randomly, which we assigned to its number of gates (four gates), we decided to use GRU which has simpler arhitecture (two gates) as it combines short-term and long-term memory in one vector.

C. *State of the art (Transformers)*

Using the attention mechanism, transformers are a direct upgrade to RNNs. We decided to use the BERT (Bidirectional Encoder Representations from Transformers) model, since it is the most widely used open-source transformer. We also used the Distilled BERT (referred to as DistilBERT) model, which is a smaller version of BERT, since a full BERT is demanding on hardware.

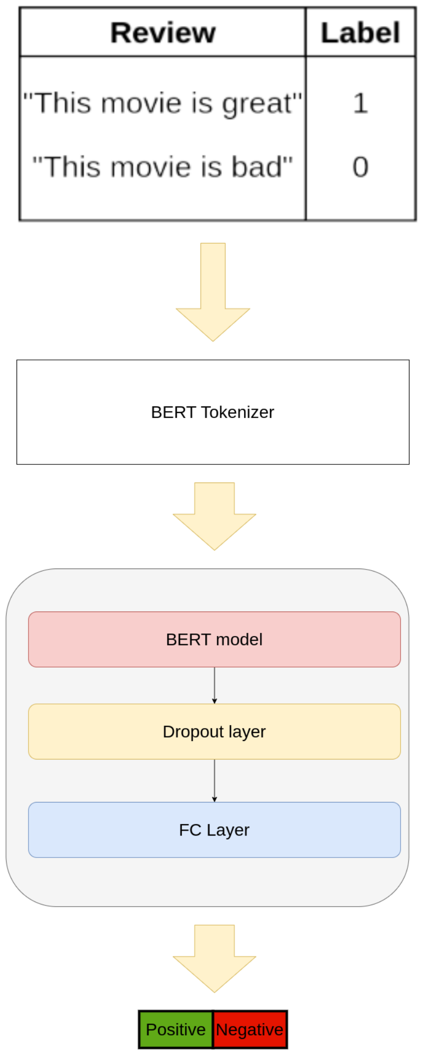

Fig. II.3. BERT model pipeline

The pipeline (see figure II.3) used to implement the BERT models differs from the others by the fact that a dictionary wasn't created. Instead, we used a pre-made tokenizer for the BERT and DistilBERT models, which turns a whole review into tokens. The maximum amount of tokens that could be used as the input was 512 tokens. Some of the reviews in dataset B exceeded that word count, so we truncated those reviews to the maximum length. This wasn't a problem since only 0.4% of data have a length longer then 512 tokens (see figure II.4) so the loss of information isn't significant.

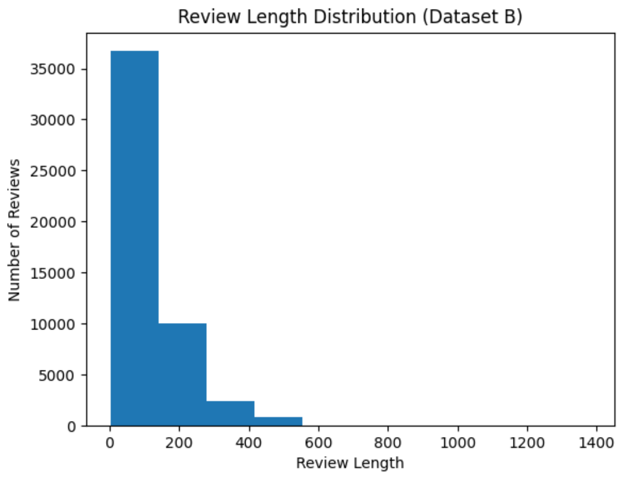

Fig. II.4. Dataset B review length

The created tokens are then used as input for the respective BERT model, we applied dropout regularization on the last layer of the BERT model, and as the output layer we used a fully connected layer with an output of two neurons.

III. CASE STUDY

A. *Datasets*

As mentioned above, in this paper we used two datasets. Both datasets consisted of user reviews taken from IMDb and their respective class, which could be positive or negative depending on the sentiment. Dataset A contained 2,000 reviews, while dataset B contained 50,000. The distribution of classes for the datasets is uniform, meaning no inherent bias in the data.

Dataset B was used in all aforementioned methods, while dataset A was limited just to classical and modern methods. The reason for this exemption was dataset A-s size; 2,000 reviews isn't enough data to feasibly train RNNs without battling overfitting, and it would be impossible to get meaningful results using a transformer.

In case of dataset A we split the data into three equal randomized folds, and used two of them for training data, while we used the remaining fold for testing, we then changed the role of each of the folds until all folds were used in testing once. With dataset B we used the same three fold idea already mentioned for classical methods, but used the standard train, validation, test split for the other methods. From the 50,000 reviews in dataset B, 40,000 were used for training, 9,000 were used for validation, and 1,000 were used for testing.

B. *Evaluation*

The evaluation metric used for every method was accuracy, which is calculated as the percentage of correct predictions with respect to all predictions. With classical methods and

RNNs trained with dataset A, accuracy was calculated as an average of all of the different test fold accuracies. The other methods used the train, validation and test split, so for those methods we used just the test set accuracy. All tests were performed on the best version of the models, which we based on validation accuracy.

IV. **RESULTS**

Model	Test accuracy
NB	78.6%
ME	84.3%
SVM	84.9%
SimpleRNN	53.0%
LSTM	79.8%
GRU	83.0%
SimpleRNN (removed outliers)	57.3%
LSTM (removed outliers)	82.4%
GRU (removed outliers)	82.9%

TABLE I

3 FOLD AVERAGE TEST RESULTS FOR ALL MODELS THAT WERE EMPLOYED ON DATASET A.

In table I results are shown for average test accuracy across 3 folds. Name of the model employed can be found in the left column, while it's results are in the right column of the same row. For RNNs results are shown when outliers were removed and when they weren't.

Model	Test accuracy
NB (3 fold avg.)	85.3%
ME (3 fold avg.)	87.8%
SVM (3 fold avg.)	85.5%
SimpleRNN	90.0%
LSTM	90.2%
GRU	90.4%
SimpleRNN (removed outliers)	90.2%
LSTM (removed outliers)	90.9%
GRU (removed outliers)	90.3%
BERT	91.0%
DistilBERT	93.5%

TABLE II

TEST RESULTS FOR ALL MODELS THAT WERE EMPLOYED ON DATASET B.

Results of test accuracy on dataset B are shown in table II. Name of the model employed can be found in the left column, while it's results are in the right column of the same row. For classical models, namely Naive Bayes, Maximum Entropy and Support Vector Machines results are shown as average of test results on all of the folds. For RNNs test results are shown for cases where outliers were removed and where they weren't. In addition to dataset A test results, on dataset B we tested BERT and DistilBERT models.

V. **DISCUSSION**

A. *Dataset A*

The results of classical methods were expected. Classical methods performed better than in the referenced paper [1]; this can be attributed to a larger dataset. As in the paper [1], naive bayes performed the poorest, that is because it assumes that each word is an independent variable, which it is not.

RNNs overall gave worse results than expected. None of the networks are better than the best classical method, maximum entropy.

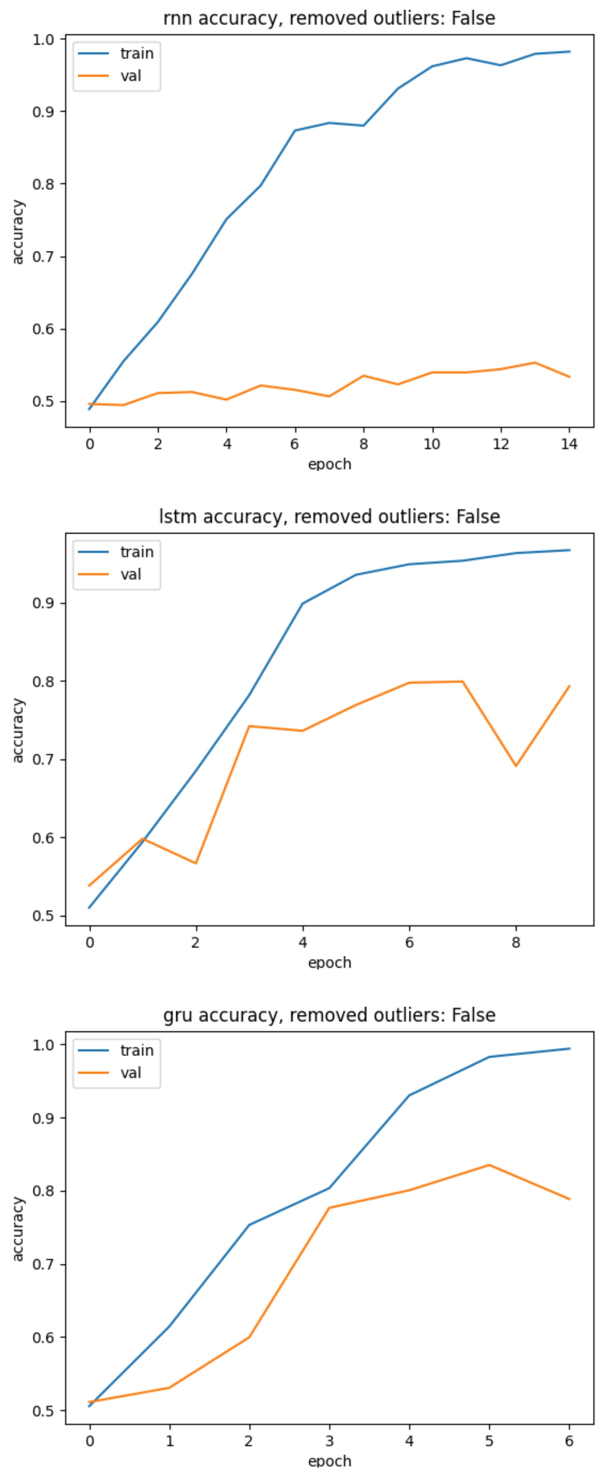

Fig. V.5. Figures of the training accuracies of all RNN models trained on dataset A without removing outliers

RNN models behaved somewhat randomly (see figure V.5). Results are the same without removing outliers. Overfitting happens quickly and that is why the validation accuracies are somewhat erratic.

B. Dataset B

Classical methods behaved better with the larger dataset. One suprise was the accuracy of the naive bayes model, which is now comparable to the other classical methods.

RNNs also performed better, they benefited from a larger dataset. RNNs now outperform the classical methods, and based on the previous observation, we can conclude that dataset A is too small to properly train a RNN model. From tables I and II we can see that Simple RNN and LSTM models benefited from removing outliers, while the GRU model is better without removing outliers.

Figures V.6 - V.11 show the training of all of the RNNs trained on dataset B.

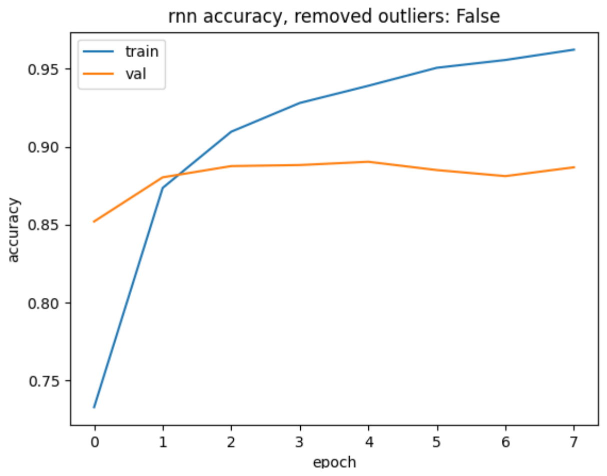

Fig. V.6. Training accuracies of the Simple RNN model trained on dataset B without removing outliers

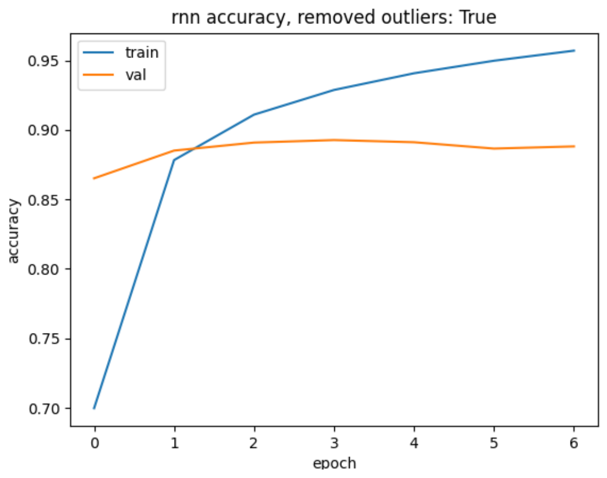

Fig. V.7. Training accuracies of the Simple RNN model trained on dataset B with removing outliers

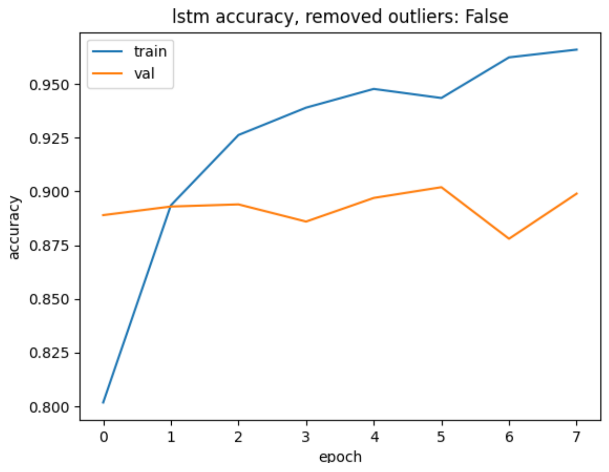

Fig. V.8. Training accuracies of the LSTM model trained on dataset B without removing outliers

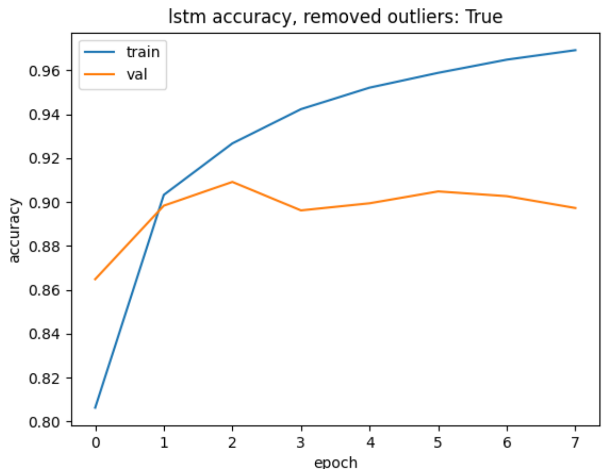

Fig. V.9. Training accuracies of the LSTM model trained on dataset B with removing outliers

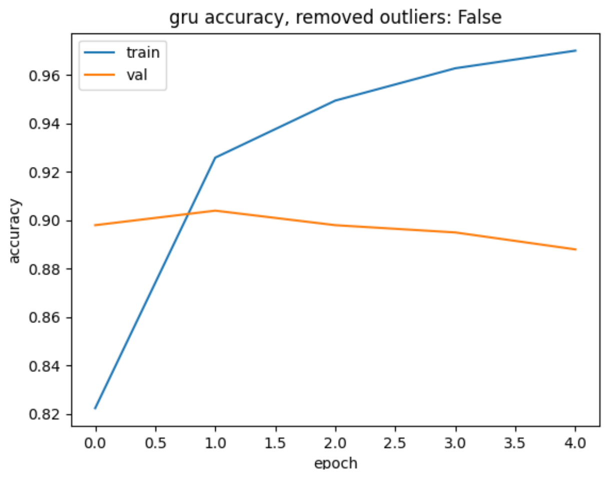

Fig. V.10. Training accuracies of the GRU model trained on dataset B without removing outliers

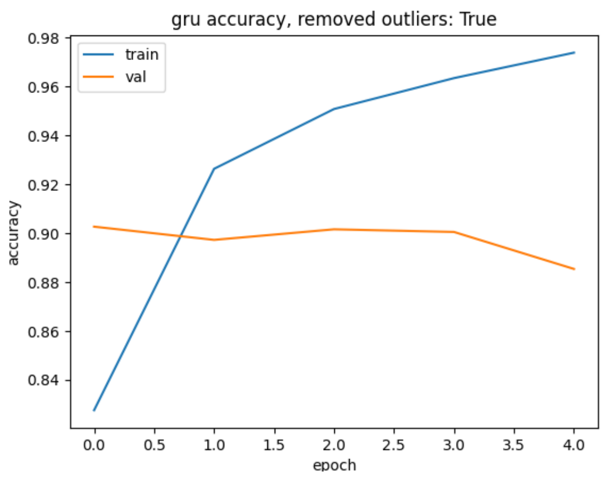

Fig. V.11. Training accuracies of the GRU model trained on dataset B with removing outliers

Even though RNNs showed better results, they still overfitted pretty quickly; meaning they would still have benefited from a larger dataset.

Transformers performed the best out of all the methods which was to be expected. From the table II: the accuracy of the BERT model was 91%, which is comparable to the LSTM model; the DistilBERT model achieved an accuracy of 93.5%, which is a significant increase in quality. The reason why DistilBert outperformed BERT is, again, the size of the dataset. Namely, even with a size of 50,000 reviews, it is still not enough to train transformer models without overfitting (figures V.12 and V.13).

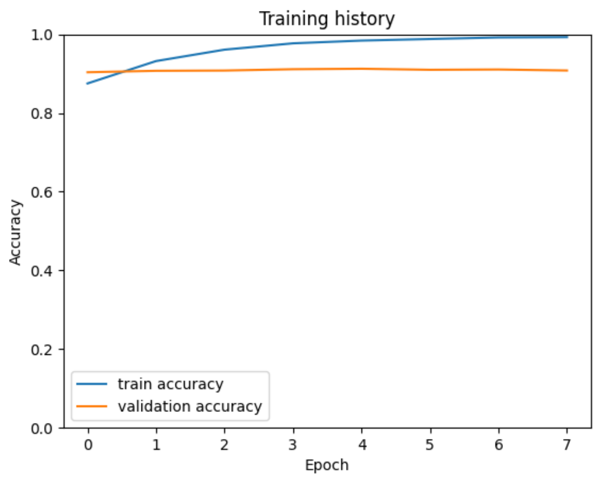

Fig. V.12. Training accuracies of the BERT model

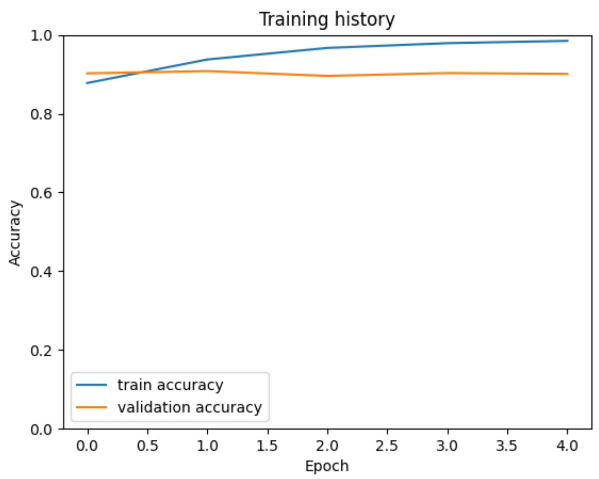

Fig. V.13. Training accuracies of the DistilBERT model

VI. Conclusion

We reimplemented the referenced paper [1] published in 2002, and tried to improve on the results. Our mission was to see how the evolution of machine learning algorithms would impact the accuracy of sentiment analysis. As shown in the Results section of this paper, we improved on the paper's results significantly, especially using dataset B, which had a significantly larger size in comparison to our baseline paper. Because of the larger dataset, we were able to use more complex machine learning models, namely BERT and DistilBERT, with transformer architecture. Dataset B also improved all of our existing models used on the smaller dataset A. We can conclude that a better and larger dataset is the key to training supervised machine learning algorithms, as it even improved the classical algorithms. We were surprised to see the great performance of classical algorithms, as their architecture isn't as complex as RNNs or Transformers. They also take significantly less time to train in comparison to RNNs and Transformers. From our results, we can see that context matters, as RNNs and Transformers that take context into consideration perform better than classical methods. For future work, BERT and DistilBERT could be improved by better fine-tuning. We had limited time, and fine-tuning would take us a lot of time because of the complex architecture and our limited computing power. Complex architectures like RNNs and Transformers could also greatly benefit from even more data as they quickly overfit training data. To better utilize such a complex architecture, the distinction between positive and negative reviews could be turned into numerical grade estimation, for example, 1 to 5 star reviews.

References

[1] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? Sentiment Classification using Machine Learning Techniques. In Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002), pages 79–86. Association for Computational Linguistics.

[2] This data was first used in Bo Pang and Lillian Lee, “A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts”, Proceedings of the ACL, 2004. (https://www.cs.cornell.edu/people/pabo/movie-review-data/)

[3] Kaggle, 25.06.2023., https://www.kaggle.com/datasets/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews