Implement Sentiment Analysis consisting of Subjectivity Detection and Polarity Classification

Mostafa Haggag (229674)

University of Trento

mostafa.haggag@studenti.unitn.it

1. Introduction

This is a report for the final project of Natural Language Understanding at the University of Trento. The main goal is to implement sentiment analysis through 2 steps where you first classify subjective and objective reviews and remove objective sentences. The second step is implementing polarity classification on the dataset containing exclusively subjective sentences. I implemented three different architectures based on gated recurrent units (GRU) for both Subjectivity and Polarity classification. Not to mention, I used embedding dropout, locked dropout, and gradient clapping to avoid overfitting my model. I implemented a Naive Bayes classifier to work as a Baseline model where I compared to it the results of the deep neural networks.

2. Task Formalisation

Sentiment analysis is the process of extracting emotions and opinions from a corpus. This task allows us to understand the attitudes opinions and emotions in the text. When using a sentiment analysis model, one can easily extract hidden information in a corpus like the user's likes and dislikes. The main goal of sentiment analysis is to determine the attitudes of a writer or speaker toward a given topic.

There are different levels of sentiment analysis where one can implement sentiment analysis at a document level, sentence level, and entity and aspect level. This report works mainly on sentiment analysis for document level where there are many different sentences and the task is to classify the category of the document whether it is a positive review or negative review. We are given a document $d \in \mathbb{X}$, where $\mathbb{X}$ is the documents space and a fixed set of sentiment classes $\mathbb{C} = c_1, x_2.....c_j$. Classes can be called labels or categories. We are trying to categorize each document to its correct label sentiment. Using the supervised deep learning method, we can learn a classifier function γ that maps a document to its correct sentiment as shown in eq.1.

$$\gamma : \mathbb{X} \rightarrow \mathbb{C} \tag{1}$$

Recurrent Neural Networks are commonly used when we are dealing with sequential data to model the function γ. The reason is, that the model uses layers that give the model a short-term memory. Using this memory, the model can predict the next data more accurately. The time for which the information about the past data will be kept is not fixed, but it depends on the weights allotted to it. Thus, RNN can be used for Sentiment Analysis.

As shown by Pong [1], one can improve the results of the sentiment analysis by removing the objective sentences in the dataset. This pre-step help achieve higher accuracy results for your final Sentiment Analysis model. This work is what I am implementing here but using Recurrent neural networks for both subjectivity detection and polarity detection. The polarity detection is to detect if the movie reviews are positive reviews or negative reviews.

3. Data Description & Analysis

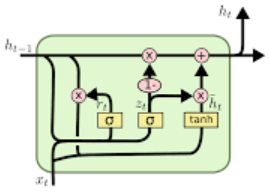
Two datasets are introduced to implement subjectivity detection and polarity detection and they are both for movie review. For both datasets, I clean the dataset by removing HTML tags, and website links and I fix most occurring appreciations in the dataset. I tried removing stopwords from both datasets but the three different model accuracies decreased after this step so I kept stopwords during training and testing.

3.1. Rotten IMBD subjectivity data-set

This dataset was first introduced by Pong [1] for detecting if the document is subjective or not. The dataset contains 5000 objective sentences and 5000 subjective sentences. The dataset is divided into a training set with 7000 sentences and a testing set with 3000 sentences. The training set is divided into 6300 for training and the rest as a validation set. The training set contains 5837 words where I set a minimum count frequency of 3 where words that occur 3 times or less are excluded from my vocab and treated as an unknown token. I have two special tokens in the vocabulary which are $< unk >, < pad >$ for unknown words and for padding the sentences to match batch size. I used this data to train and test the performance of an objective classifier.

3.2. Rotten IMBD subjectivity data-set

The other dataset is made of large movie reviews from IMBD prepared by Maas [2] containing 12500 negative reviews and 12500 positive reviews. The total dataset is consistent with 50000 reviews and they are divided into 25000 for training and 25000 for testing. I take this dataset and segment each sentence in every review and input it into the subjectivity detector to detect if this sentence is subjective or not and remove this sentence in case it is objective. This can lead to a complete document entry being removed because these reviews are objective. The objectivity classifier removed 51 reviews completely from the training set and 35 reviews from the test set. I divide my data into 19959 reviews for the training set, 4990 for the validating set, and 24965 for the testing set including special tokens. The training set contains 31416 words where I set a minimum count frequency of 3. I used this data to train and test the performance of a Subjectivity classifier.



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Figure 1: *As seen in the figure we can see the GRU unit with the forget gate and output gate on the right and on the left the equations presenting how the gru computes the next hidden state*

4. Model

4.1. Tokenizer

I use **en_core_web_lg** English model to tokenize the words of the sentences however I use sentencizer pipeline in spacy to detect boundaries of sentences without dependency parsing. The main reason why I use this method is that it runs faster compared to sentence segmentation performed by the Dependency-Parser pipeline in spacy and the effect on the model metrics was not very effective.

4.2. Word embeddeding

I choose to work with GloVe [3] to represent words in my vocabulary. The main reason behind this choice for this embedding is because it derived from semantic relationships between words from the co-occurrence matrix capturing both local and global statistics of corpus which is a big advantage when compared to the word2vec method. I use the pre-trained word vectors trained on Wikipedia 2014 with 6 billion tokens with a dimension of 100D.

4.3. GRU Models

Introduction about GRU The Gated Recurrent Unit is the simpler variation of the LSTM (Long-Short Term Memory). Both these architectures were proposed as the remedy for the RNN's vanishing gradient problems. GRU comprised only two gates, while its ancestor - LSTM had 3 gates in its unit is much simpler to compute and implement and it has been able to prevent vanishing gradient than LSTMS.

The GRU consists of two gates, the update gate which selects whether the hidden state is to be updated with a new hidden state, and the Reset gate which decides whether the previous hidden state is ignored. As seen in Figure1, we see a gru unit. All the models are using the same loss funciton which is Cross entropy loss and use adam optimizer.

4.3.1. GRU Model

This model was with 3 layers of bidirectional GRU with 256 set as the hidden dimension for both subjectivity and polarity classifier. The hidden state of the last gru unit in the sequence for the last layer in both directions is taken and concatenated and fed to a fully connected layer that is used to predict the label.

4.3.2. GRU with padding

This model was inspired by [4] where the model is made of 3 layers of bidirectional GRUs with 256 set as the hidden dimension for both classifiers. The hidden states for the last unit in the last layer in both directions are added together into 1 tensor. All the hidden states in the sequence for the last layer are added together in both directions. We use this output to select the maximum value over each dimension of the hidden representation using maximum pooling and to do mean pooling by considering the average of the representation by summing along the batch axis and dividing by the length. The output of the maximum pooling, mean pooling, and summed last layer units are concentrated and fed to the linear layer with a softmax activation function to predict the output. The main reason behind this is that documents consist of hundreds of words and information can get lost if we get only the last hidden state of the model. For this reason, I concatenate the hidden state at the last time in step h_T of the document with both the max pooled and mean pooling representation of the hidden state with a number of steps in the sequence. This is shown in eq.2.

$$h_c = [H_T, maxpool(H), meanpool(H] \tag{2}$$

4.3.3. GRU with attention

This model was inspired by [5] where the model is made of 3 layers of bidirectional GRUs with 256 sets as the hidden dimension for both classifiers. I use an attention mechanism where I have a matrix H consisting of output vector$[h_1, h2, h_3....h_T]$ that GRU unit produced with T the sequence length. The representation r of the sentence is formed by a weighted sum of these output vectors:

$$M = \tanh H \tag{3}$$

$$\alpha = softmax(w^T M) \tag{4}$$

$$r = H\alpha^T \tag{5}$$

I obtain the final sentence pair representation using this equation and I feed it to a linear layer to predict the output.

$$h^* = \tanh r \tag{6}$$

4.4. Regularization Techniques

Regularization Techniques are very important for GRU because of the exploding and vanishing gradients problem.

4.4.1. Embedding dropout

This approach was introduced by Gal [6] where it is performing dropout on the embedding matrix at a word level by dropping inputs words which helps in regularizing the embedding layer. The remaining non-dropped word embedding is scaled by $\frac{1}{1-p_c}$ where p_c is the probability of dropout of the embedding layer.

4.4.2. Locked droppout

I use the same dropout mask at each time step where I drop the same units at the same time step. This is different from normal dropout where the mask for each time step is different. I use it after the embedding dropout because I want to lock the dropout mask for same words through different steps.

4.4.3. Gradient clipping

This technique is used to prevent the exploding gradients where we clip the computed gradients into a certain range to have control over the gradient update iteration. You are mainly changing the error derivative before propagating it back through the network and using it to update the weights.

5. Evaluation

5.1. Metrics

I use F1 score to evaluate my model. The F1 score is defined as the harmonic mean of precision and recall. In the F1 score, we compute the average of precision and recall. They are both rates, which makes it a logical choice to use the harmonic mean. The F1 score equation is seen in eq.7.

$$F_1 = 2 * \frac{Preccision * Recall}{Preccision + Recall} \tag{7}$$

5.2. Subjectivity classifier

I started fine-tuning the parameters of the 3 models where I evaluated the models according to the F1 score. The embedding dropout is considered the game changer technique that helped in regularizing the 3 models. I used the validation set to check that the model is not overfitting. I used to extract the false positive and false negative examples and tried to understand what can be done to help the model to predict this document correctly. This was mainly done by the pre-cleaning step of fixing abbreviations. I tested removing stopwords and lemmatizing the words but the worse performance was achieved. The main reason why this was happening is that you lose valuable information that can help your neural network figure things out. Not to mention, Glove embedding is covering 98% of the vocabulary of the training set. I also checked the length of the longest sequence in the dataset and I found that most of the sentences in this dataset were short where the maximum token per view never surpassed 250 tokens.

I ran each model 5 times to make sure that the results I got are not due to random initialization. As seen in Figure2, We can see the Gru with pooling was performing better than the other models. This is due to the fact that we are not only using the output of the last GRU unit in the sequence but all averaging and max pooling all the outputs. I was able to achieve F1 accuracy of 94.2% and I use this model to clean the objective sentences from the Polarity dataset. As seen in Figure3, this model was able to achieve the least number of false positives and false negatives. As seen in table 1, we can see the comparison between the 3 models versus the baseline.

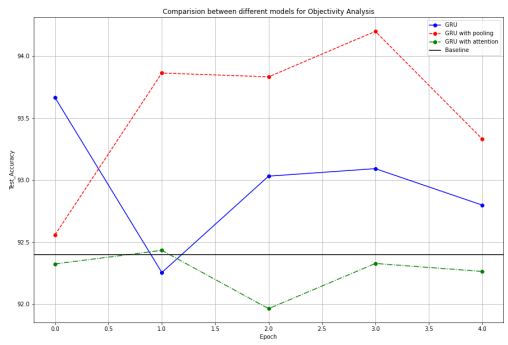


Figure 2: *As seen in the figure, we see the F1 score for different models which are run 5 times to make sure that the results we are achieving are not due to random initialization.*

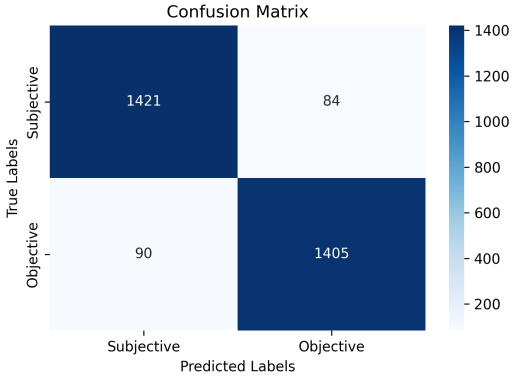


Figure 3: *As seen in the figure, we can see the confusion matrix for the best model. In blue are the examples correctly classified and in white are the example miss classified.*

Subjectivity			
Model	**F1**	**Accuracy**	**Loss**
Baseline	92%	92%	-
GRU	93.66	93.67	0.001519
GRU with pooling	94.20%	94.20%	0.001342
GRU with attention	92.26%	92.26%	0.001637

Table 1: *As seen in the table, the 3 models are overachieving compared to the Baseline. However, GRU with pooling is achieving the highest F1 accuracy. These results are the best results that each model achieved in the 5 runs*

5.3. Polarity classifier

After removing the objective sentences, I prepared the data to be input into the 3 models to do polarity classification. Removing stop-words and lemmatizing the tokens achieved negative results similar to what was happening with the objectivity dataset. The main challenge with this dataset is that some reviews are very long so I had to keep only 200 tokens as the maximum for each review. Not to mention, removing punctuation helped in decreasing the number of tokens. Due to the small gain of the GRU with attention, I decided to exclude this model from the polarity classification. The models were both trained and fine-tuned to achieve the best F1 results. seen in Figure4, We can see the Gru was performing better than the other models. I was able to achieve F1 accuracy of 89.189% and I use this model to clean the objective sentences from the Polarity dataset. As seen in Figure5, this model was able to achieve the least number of false positives and false negatives. As seen in table 1, we can see the comparison between the 2 models versus the baseline. We can see that Gru with pooling achieves very similar results in 1 of the results but on average the GRU only is able to achieve better results.

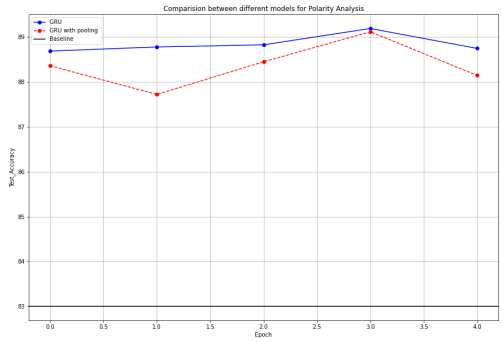


Figure 4: *As seen in the figure, we see the F1 score for different models which are run 5 times to make sure that the results we are achieving are not due to random initialization.*

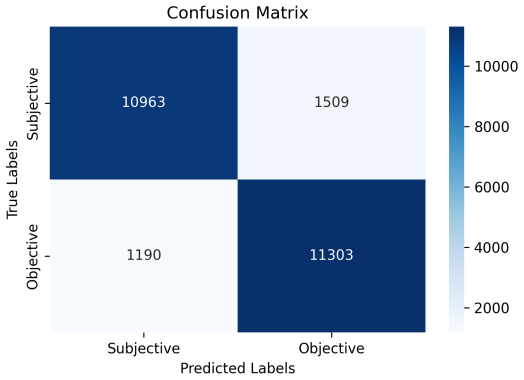


Figure 5: *As seen in the figure, we can see the confusion matrix for the best model. In blue are the examples correctly classified and in white are the example miss classified.*

Polarity			
Model	**F1**	**Accuracy**	**Loss**
Baseline	92%	92%	-
GRU	89.18%	89.19%	0.00213
GRU with pooling	89.15%	89.11%	0.00232

Table 2: *As seen in the table, the 2 models are overachieving compared to the Baseline. However, GRU is achieving the highest F1 accuracy. These results are the best results that each model achieved in the 5 runs*

6. Conclusion

As seen in the results we were able to overcome the baseline model for both the objectivity and polarity classification. More work has to be done to analyze the terrible performance behind the GRU with attention to understanding the reason behind the bad performance of the model.

7. References

[1] B. Pang and L. Lee, "A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts," *CoRR*, vol. cs.CL/0409058, 2004. [Online]. Available: http://arxiv.org/abs/cs.CL/0409058

[2] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Portland, Oregon, USA: Association for Computational Linguistics, June 2011, pp. 142–150. [Online]. Available: http://www.aclweb.org/anthology/P11-1015

[3] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543. [Online]. Available: http://www.aclweb.org/anthology/D14-1162

[4] J. Howard and S. Ruder, "Fine-tuned language models for text classification," *CoRR*, vol. abs/1801.06146, 2018. [Online]. Available: http://arxiv.org/abs/1801.06146

[5] Z. Wang and B. Yang, "Attention-based bidirectional long short-term memory networks for relation classification using knowledge distillation from bert," in *2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, 2020, pp. 562–568.

[6] Y. Gal and Z. Ghahramani, "A theoretically grounded application of dropout in recurrent neural networks," 2015. [Online]. Available: https://arxiv.org/abs/1512.05287