NCSR Demokritos & University of Peloponnese

SUBJECT

Deep Learning

Sentiment Analysis on IMDB Reviews

Authors:
Stasinos Panagiotis (dit2123)

Instructor:
Giannakopoulos Theodoros
Assignment 1

NATIONAL CENTRE FOR
SCIENTIFIC RESEARCH "DEMOKRITOS"

July 6, 2022

Introduction

The modern era has brought rapid growth to the internet and its various applications. The internet, now as a virtual interactive space, gives us the ability to share and also receive a large amount of information, affecting multiple aspects of our lives, while still having direct consequences in marketing and communication. Social media, as one of the main results of the new internet age, has the ability to manipulate consumers by shaping their opinions and attitudes towards products and situations. Through various techniques of data mining and analysis , we are now given the possibility to process to a high degree the information we draw from the numerous databases. This information results from the ability of users to publish texts, make comments and receive responses from other users. The creation of automated models that would make the information immediately usable and user-friendly has been a need but also a challenge for the scientific community. In this context, the field of Sentiment Analysis (Similar terms: mining of opinion, artificial intelligence of emotion) was created, which is experiencing remarkable growth, especially in its application to social media. It extracts useful information and patterns to predict behavior , of developments and events taking place in social media. The different types of social media include various social networking pages, blogs, video sharing pages and company networks such as Facebook, Twitter, Instagram , email interactions in a system review sites like IMDB and rotten tomatoes to know about the cast and crew of the movie, review, and ratings. The prediction and analysis of information that these domains offers us ,includes multiple application areas such as marketing, economics, politics and health.

Sentiment Analysis

Sentiment Analysis, also known as Opinion Mining, is a field that studies and analyzes the feelings, opinions and attitudes of people, expressed through written words, towards an entity. The entity can be a product, some service, people, organizations, events, issues and topics. Sentiment analysis used natural language processing (NLP) to obtain desired information from concerned resources. Sentiment analysis is mostly used for the reviewing of various products in different application domains, for instance market research and customer service, etc [1]. For sentiment categorization, the efficiency of the sentiment lexicons at the sentence level and document level was evaluated using a news headlines dataset and a dataset of Amazon product review [2]. The goal of sentiment analysis is to implement automatic tools capable of extracting subjective information from natural language texts, such as opinions and emotions, so as to create a structured and actionable knowledge to be used either as a decision support system or as an individual decision [3]. In short, sentiment analysis or opinion mining aims to identify positive and negative opinions or sentiments expressed or implied in a text and also the entities targeted by those opinions or sentiments.

Purpose of this project

The purpose of this work is the approach and implementation of a sensitivity analysis system, which will present the polarity (positive, negative) of opinions that have been drawn through the infamous IMDB Reviews dataset from Kaggle. This dataset contains a total of 50.000 reviews on various movies and it is devided equally on both positive and negative reviews (25.000 each). To create the desired model we trained multilayered perceptrons on a database of IMDB movie review by Stanford which gives around 88% accuracy while testing and around 96% of accuracy when training

Methodology

In this section we will present our methodology in steps as a way of representation in a notebook.

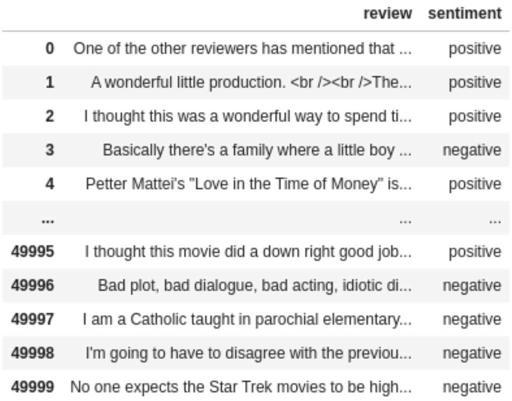
- **Main Libraries**

  We start by importing the main libraires that we will use:

  1.the re module (for regular expression matching operations)
  2.the nltk toolkit (for natural language operations)
  3.the random module (for random number generation)
  4.the numpy library (for arrays operations)
  5.the pandas library (for data analysis)
  6.the scipy.stats module (for statistics)
  7.the seaborn library (for statistical data visualization)
  8.the matplotlib.pyplot interface (for MATLAB-like plots)
  We also download the stopwords and punkt data packages from the nltk toolkit.

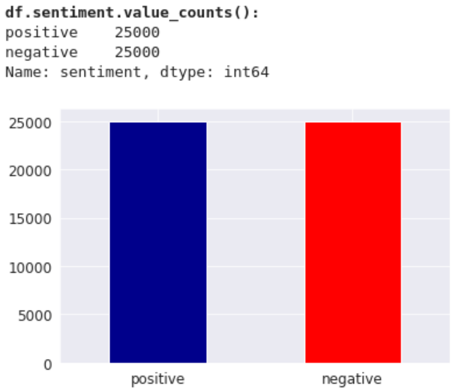
- **Dataset**

  As mentioned before the dataset we are going to use is the IMDB dataset from Kaggle, which consists of 50.000 reviews.Dataset selection is important in order to acquire or create an extensive database. We want it to contain a sufficient amount of data so that we can get the best possible results. In most cases the data set will also contain redundant information also called noise, however this problem is dealt with in the pre-processing stage with various techniques. In “Fig. 1” & “Fig. 3” we can see that our dataset is equally devided into negative and positive reviews.



	review	sentiment
0	One of the other reviewers has mentioned that ...	positive
1	A wonderful little production. <br /><br />The...	positive
2	I thought this was a wonderful way to spend ti...	positive
3	Basically there's a family where a little boy ...	negative
4	Petter Mattei's "Love in the Time of Money" is...	positive
...	...	...
49995	I thought this movie did a down right good job...	positive
49996	Bad plot, bad dialogue, bad acting, idiotic di...	negative
49997	I am a Catholic taught in parochial elementary...	negative
49998	I'm going to have to disagree with the previou...	negative
49999	No one expects the Star Trek movies to be high...	negative

Figure 1: IMDB Dataset



```
df.sentiment.value_counts():
positive    25000
negative    25000
Name: sentiment, dtype: int64
```

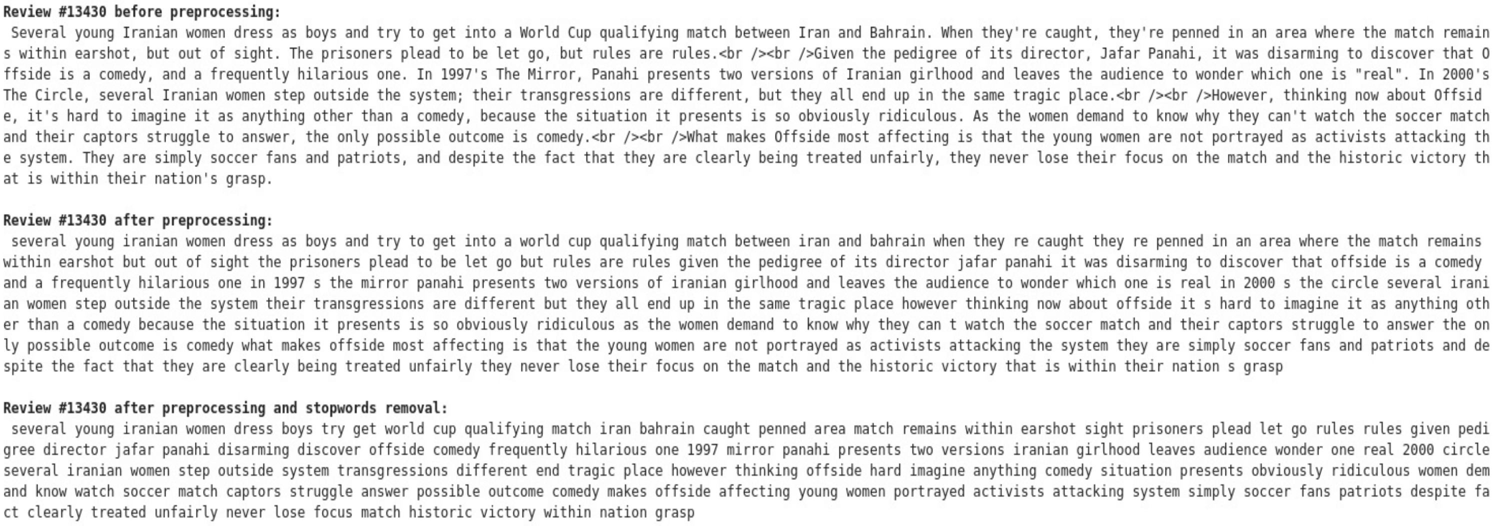
Figure 2: Distribution of the Dataset

- **Data Preprocessing**

  Data pre-processing is done to eliminate incomplete, noisy or inconsistent information. It is an important and mandatory step, which is performed first when using any of the supervised machine learning algorithms. To extract the useful information from a text on twitter it is critical to remove some of the building blocks of a tweet. These are the:
  1.punctuation removal
  2.HTML tags removal
  3.URL’s removal
  4.removal of characters which are not letters or digits
  5.removal of successive whitespaces
  6.transformation of the text to lower case
  7.whitespaces strip from the beginning and the end of the reviews

  A better way to represent these steps is to actually show how they can transform a text. In the following figure , “Fig. 3” , we have printed a random review from our dataset. This figure presents the three steps of the cleaning phase, starting from an uncleaned text and finally, after performing the 7 steps mentioned above, we result in the desired form [4].

  Moreover we use LabelEncoder in order to transform every string value ("Positive","Negative") into numerical values, for this case 0 and 1 since we only have to possible outcomes.



```
Review #13430 before preprocessing:
 Several young Iranian women dress as boys and try to get into a World Cup qualifying match between Iran and Bahrain. When they're caught, they're penned in an area where the match remain
s within earshot, but out of sight. The prisoners plead to be let go, but rules are rules.<br /><br />Given the pedigree of its director, Jafar Panahi, it was disarming to discover that O
ffside is a comedy, and a frequently hilarious one. In 1997's The Mirror, Panahi presents two versions of Iranian girlhood and leaves the audience to wonder which one is "real". In 2000's
The Circle, several Iranian women step outside the system; their transgressions are different, but they all end up in the same tragic place.<br /><br />However, thinking now about Offsid
e, it's hard to imagine it as anything other than a comedy, because the situation it presents is so obviously ridiculous. As the women demand to know why they can't watch the soccer match
and their captors struggle to answer, the only possible outcome is comedy.<br /><br />What makes Offside most affecting is that the young women are not portrayed as activists attacking th
e system. They are simply soccer fans and patriots, and despite the fact that they are clearly being treated unfairly, they never lose their focus on the match and the historic victory th
at is within their nation's grasp.

Review #13430 after preprocessing:
 several young iranian women dress as boys and try to get into a world cup qualifying match between iran and bahrain when they re caught they re penned in an area where the match remains
within earshot but out of sight the prisoners plead to be let go but rules are rules given the pedigree of its director jafar panahi it was disarming to discover that offside is a comedy
and a frequently hilarious one in 1997 s the mirror panahi presents two versions of iranian girlhood and leaves the audience to wonder which one is real in 2000 s the circle several irani
an women step outside the system their transgressions are different but they all end up in the same tragic place however thinking now about offside it s hard to imagine it as anything oth
er than a comedy because the situation it presents is so obviously ridiculous as the women demand to know why they can t watch the soccer match and their captors struggle to answer the on
ly possible outcome is comedy what makes offside most affecting is that the young women are not portrayed as activists attacking the system they are simply soccer fans and patriots and de
spite the fact that they are clearly being treated unfairly they never lose their focus on the match and the historic victory that is within their nation s grasp

Review #13430 after preprocessing and stopwords removal:
 several young iranian women dress boys try get world cup qualifying match iran bahrain caught penned area match remains within earshot sight prisoners plead let go rules rules given pedi
gree director jafar panahi disarming discover offside comedy frequently hilarious one 1997 mirror panahi presents two versions iranian girlhood leaves audience wonder one real 2000 circle
several iranian women step outside system transgressions different end tragic place however thinking offside hard imagine anything comedy situation presents obviously ridiculous women dem
and know watch soccer match captors struggle answer possible outcome comedy makes offside affecting young women portrayed activists attacking system simply soccer fans patriots despite fa
ct clearly treated unfairly never lose focus match historic victory within nation grasp
```

Figure 3: Preprocessing steps

- **Data splitting and Tokenization**
  We want to have a sufficient number of data in order to train our model but also we need to test the performance of our model. For that reason we split our dataset into a train dataset containing 70% of the total data and a test dataset containg 30% of the data. Next, we use the Tokenizer class from keras.preprocessing.text module to create a dictionary of the "dict_size" most frequent words present in the reviews (a unique integer is assigned to each word).Tokenization is the process of breaking a stream of text into words, phrases, symbols, or other meaningful elements called tokens .The aim of the tokenization is the exploration of the words in a sentence. The list of tokens becomes input for further processing such as parsing or text mining. [5] The index of the Tokenizer is computed the same way no matter how many most frequent words we use later. In “Fig. 4” & “Fig. 5” we can see which are the most used words that appear in the positive and negative reviews respectively, via the help of WordCloud.

Positive Sentiment
film
br
movie
one
character
really
way
show
even
good
time
see
story
great
make
much
well
love
made
still
people
think
life
work
know
watch
best
seen
find
look

Figure 4: WordCloud- Positive Words

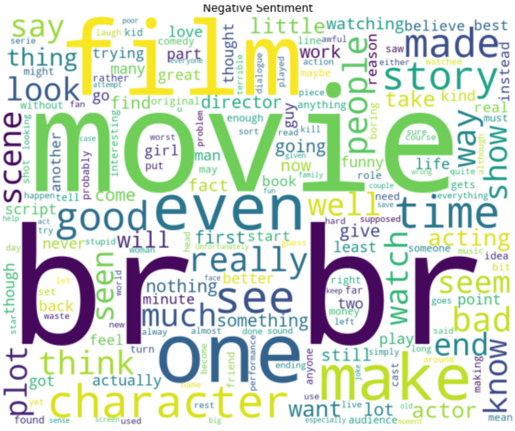


Figure 5: WordCloud- Negative Words

Afterwards, We use the texts_to_sequences() function of the Tokenizer class to convert the training reviews and test reviews to lists of sequences of integers (tokens) "train_rev_tokens" and "test_rev_tokens".If the lengths of the sequences were normally distributed, then a given length could be considered small or large when outside the interval,

$$mean\_value\_of\_seq\_lengths \pm 2\_standard\_deviations\_of\_seq\_lengths$$

and lengths not belonging to this interval would only represent 5% of the elements of seq_lengths [6] . Here, we follow this heuristics, and thus define an upper bound for the length of sequences accordingly.

A more explanatory example can be viewed in “Fig. 6”.

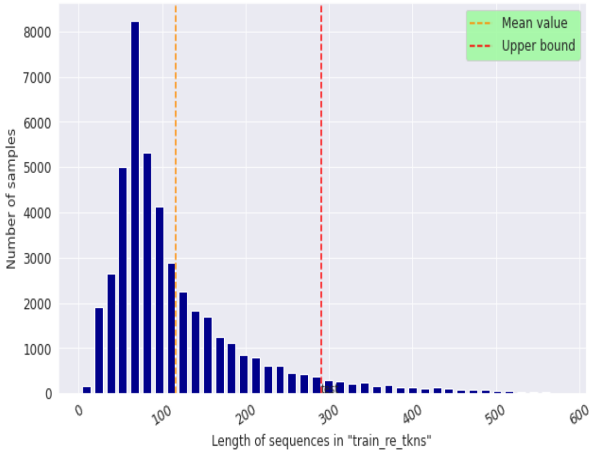
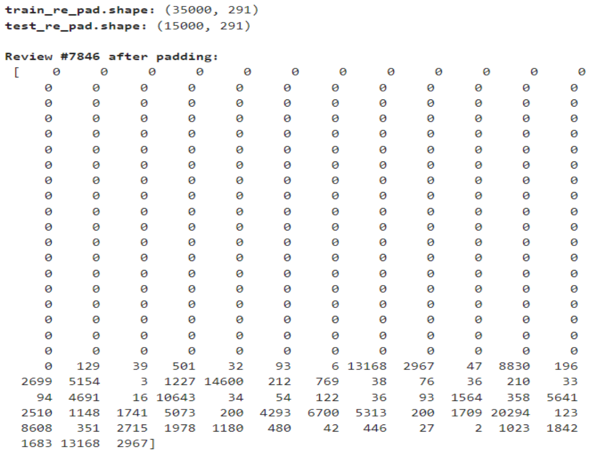


Figure 6: Chosen upper bound for the length of sequences

- **Preparing our data for the Neural Network**
  In Keras, we create neural networks either using function API or sequential API. In both the APIs, it is very difficult to prepare data for the input layer to model, especially for RNN and LSTM models. This is because of the varying length of the input sequence. The variable lengths of the input sequence of data need to be converted to an equal length format. This task is achieved using masking or embeddings and padding in Keras or TensorFlow. Padding in Keras reshape the variable length input sequence to sequence of the same length [7].Using the pad_sequences() function from keras_preprocessing.sequence module, we transform "train_re_tkns" and "test_re_tkns" into 2D numpy arrays of shape (number of sequences, upper). Sequences of length smaller (resp. larger) than "upper" are extended (resp. truncated) to get a length equal to "upper". A sample of what we have been describing in this step can be seen in “Fig. 7”, where we have printed the padded version of a random review.



```
train_re_pad.shape: (35000, 291)
test_re_pad.shape: (15000, 291)

Review #7846 after padding:
 [    0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0   129    39   501    32    93     6 13168  2967    47  8830   196
  2699  5154     3  1227 14600   212   769    38    76    36   210    33
    94  4691    16 10643    34    54   122    36    93  1564   358  5641
  2510  1148  1741  5073   200  4293  6700  5313   200  1709 20294   123
  8608   351  2715  1978  1180   480    42   446    27     2  1023  1842
  1683 13168  2967]
```

Figure 7: Padded Sequences

The concept of masking is that we can not train the model on padded values. The placeholder value subset of the input sequence can not be ignored and must be informed to the system. This technique to recognize and ignore padded values is called Masking in Keras. One way of performing the Masking is via the help of an Embedding Layer. An embedding layer is a word embedding that is learned jointly with a neural network model on a specific NLP task. It requires text to be preprocessed such that each word is one-hot encoded. The size of the vector space is specified as part of the model usually 50, 100, 200 or 300 dimensions, initialized with small random numbers. The embedding layer is used on the front end of a neural network and is fit in a supervised way using the Backpropagation algorithm. The one-hot encoded words are mapped to the word vectors. [8]

- **Neural Network**
  After a lot of experimentation with various layers and parameters we have concluded into a model that consists of one Conv1D layer and two LSTM layers. We also have added two Dropout layers in order to reduce the overfitting that was observed during the training phase, these Dropout layers were inserted due to the fact that the validation loss of our model was diverging from the loss in our training set. A better look of our model can be seen in "Fig. 8"

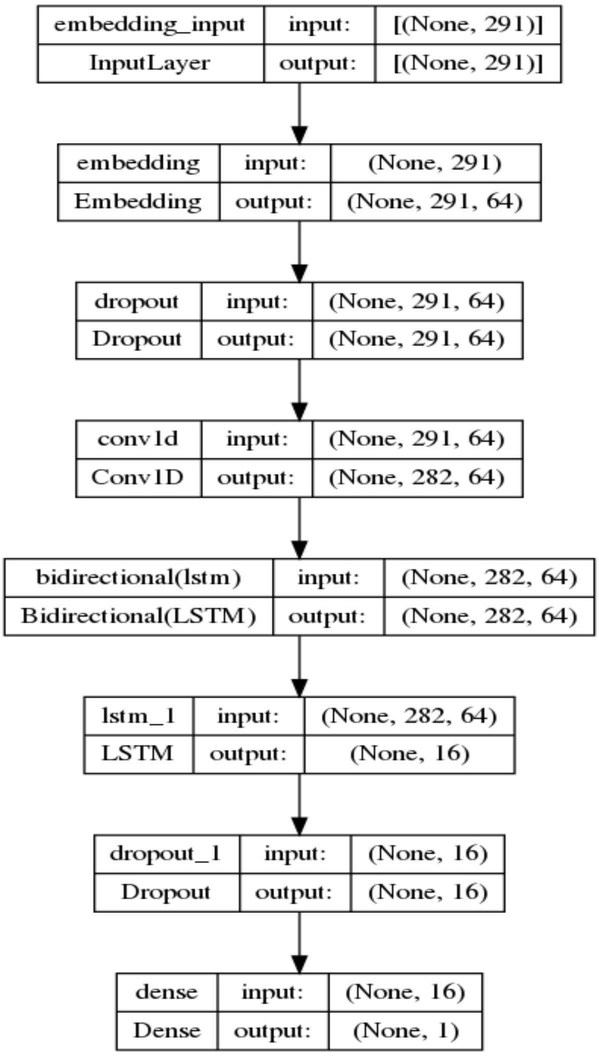
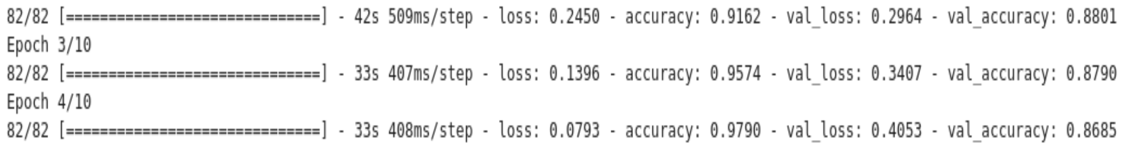

embedding_input | input: | [(None, 291)]
InputLayer | output: | [(None, 291)]
embedding | input: | (None, 291)
Embedding | output: | (None, 291, 64)
dropout | input: | (None, 291, 64)
Dropout | output: | (None, 291, 64)
conv1d | input: | (None, 291, 64)
Conv1D | output: | (None, 282, 64)
bidirectional(lstm) | input: | (None, 282, 64)
Bidirectional(LSTM) | output: | (None, 282, 64)
lstm_1 | input: | (None, 282, 64)
LSTM | output: | (None, 16)
dropout_1 | input: | (None, 16)
Dropout | output: | (None, 16)
dense | input: | (None, 16)
Dense | output: | (None, 1)

Figure 8: Neural Network

- **Training Phase**
  In the training phase we get our model to train in five epochs. A valuable tool while training a neural network is callbacks. Callbacks are really helpful as they stop our model when the validation accuracy of our model starts decreasing for consecutive 2 epochs as well save the best possible weights which gives highest validation accuracy. As we can see in the following figure "Fig. 9", in the 3rd epoch the validation accuracy of our model started deteriorating and in the 4th epoch it didnt improve, hence our model stopr training.



```
82/82 [==============================] - 42s 509ms/step - loss: 0.2450 - accuracy: 0.9162 - val_loss: 0.2964 - val_accuracy: 0.8801
Epoch 3/10
82/82 [==============================] - 33s 407ms/step - loss: 0.1396 - accuracy: 0.9574 - val_loss: 0.3407 - val_accuracy: 0.8790
Epoch 4/10
82/82 [==============================] - 33s 408ms/step - loss: 0.0793 - accuracy: 0.9790 - val_loss: 0.4053 - val_accuracy: 0.8685
```

Figure 9: Callbacks

. A better option to present these stats is by actually plotting them. As a result in “Fig. 10” and “Fig. 11” we can have a better look on how our model is doing.

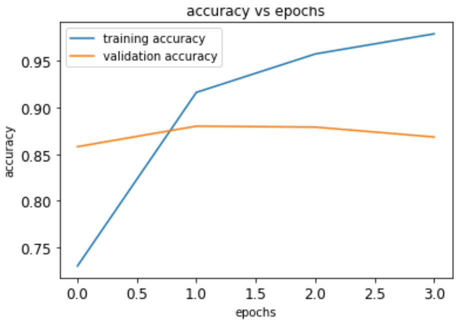


Figure 10: IMDB Dataset

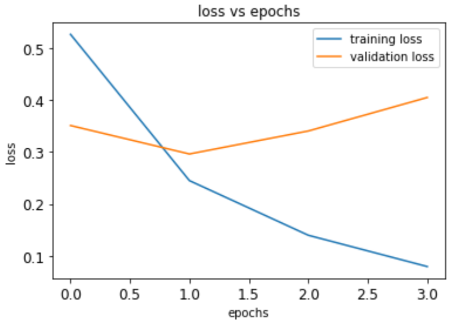
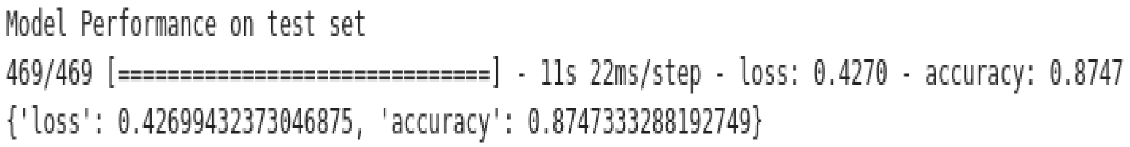


Figure 11: Distribution of the Dataset

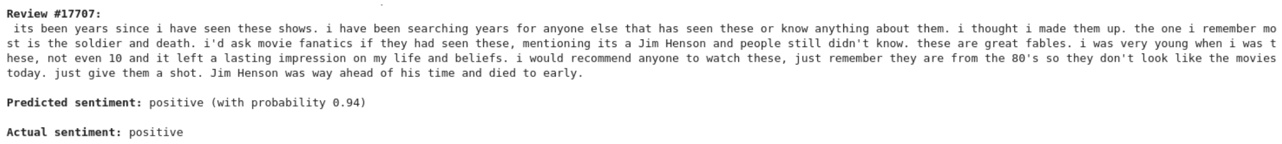
- **Results**
  First, we evaluate the loss and accuracy of the trained model on the test set “Fig. 12”. It is noticeable that the accuracy is good enough on our validation set while not having that high validation loss, which means that our model is doing good.



```
Model Performance on test set
469/469 [==============================] - 11s 22ms/step - loss: 0.4270 - accuracy: 0.8747
{'loss': 0.42699432373046875, 'accuracy': 0.8747333288192749}
```

Figure 12: Evaluation

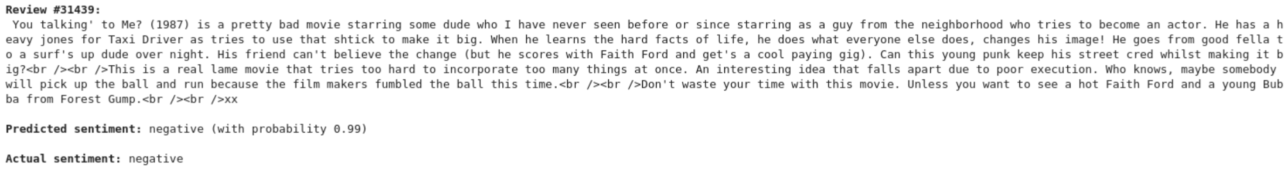
To be more accurate we will print some random reviews from out dataset, that can be seen in “Fig. 13 & “Fig. 14



```
Review #17707:
 its been years since i have seen these shows. i have been searching years for anyone else that has seen these or know anything about them. i thought i made them up. the one i remember mo
st is the soldier and death. i'd ask movie fanatics if they had seen these, mentioning its a Jim Henson and people still didn't know. these are great fables. i was very young when i was t
hese, not even 10 and it left a lasting impression on my life and beliefs. i would recommend anyone to watch these, just remember they are from the 80's so they don't look like the movies
today. just give them a shot. Jim Henson was way ahead of his time and died to early.

Predicted sentiment: positive (with probability 0.94)

Actual sentiment: positive
```

Figure 13: Predicted Positive Review

It can be noticed that this review is positive with a high probability of 94%, the possibility could be even higher but there are not so many words that highlight a positive sentiment except from the phrase "i would recommend".

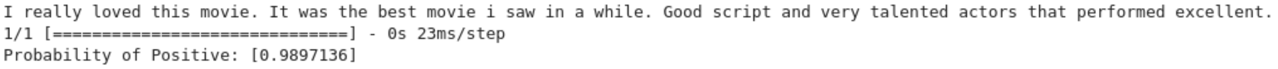


```
Review #31439:
 You talking' to Me? (1987) is a pretty bad movie starring some dude who I have never seen before or since starring as a guy from the neighborhood who tries to become an actor. He has a h
eavy jones for Taxi Driver as tries to use that shtick to make it big. When he learns the hard facts of life, he does what everyone else does, changes his image! He goes from good fella t
o a surf's up dude over night. His friend can't believe the change (but he scores with Faith Ford and get's a cool paying gig). Can this young punk keep his street cred whilst making it b
ig?<br /><br />This is a real lame movie that tries too hard to incorporate too many things at once. An interesting idea that falls apart due to poor execution. Who knows, maybe somebody
will pick up the ball and run because the film makers fumbled the ball this time.<br /><br />Don't waste your time with this movie. Unless you want to see a hot Faith Ford and a young Bub
ba from Forest Gump.<br /><br />xx

Predicted sentiment: negative (with probability 0.99)

Actual sentiment: negative
```

Figure 14: Predicted Negative Review

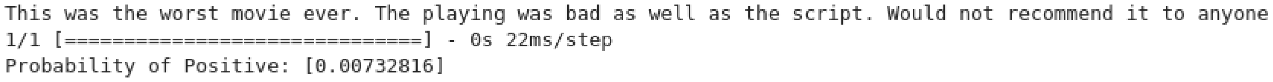
In Fig. 15 a negative review can be printed where words like "pretty bad" and "don’t waste your time" make it obvious, hence a possibility of 99% of being negative is assigned to this review.

Let’s make things more intrested. We can create our own reviews and have our model predict their sentiment. We will illustrate this with two examples: In Fig. 16 we enter the following sentence: "’I really loved this movie. It was the best movie i saw in a while. Good script and very talented actors that performed excellent.’" and we get the result that it’s positive with a possibility of 98%. On the other hand, we enter the sentence "'This was the worst movie ever. The playing was bad as well as the script. Would not recommend it to anyone" and we get a possibility of 0.007 of being positive meaning it's negative, which is what we would expect for a sentence like this.



```
I really loved this movie. It was the best movie i saw in a while. Good script and very talented actors that performed excellent.
1/1 [==============================] - 0s 23ms/step
Probability of Positive: [0.9897136]
```

Figure 15: Toy Positive Review



```
This was the worst movie ever. The playing was bad as well as the script. Would not recommend it to anyone
1/1 [==============================] - 0s 22ms/step
Probability of Positive: [0.00732816]
```

Figure 16: Toy Negative Review

Overall our model seems to work good on our own examples, which is what we wanted.

- **Conclusion and Future Work**

  In this project we tackled the problem of Sentiment Analysis, which is a widely studied subject in the field of Natural Language Processing. The dataset used for this project was the IMDB movies reviews with 25.000 thousand negative and equal size of positive reviews. First we processed our data doing some basic cleaning on the reviews and afterwards we splitted our dataset, having 0.3 of our total dataset as a test set. Then extracted the tokens from each review and we created sequences from these tokens to transform them into integers, since a computer cannot read words from sentences. We have setted an upper bound for our sentences and padded them in order to prepare them as an input for our Neural Networks.

  Our Neural Network was build and trained using an Embedding layer, while in the training phase we used callbacks with a patience of 2 in order to have the best possible validation accuracy. We end up with a validation accuracy of 87.47%. Last but not least, we test our model not only on some random reviews but also in some random sentences that we have written on our own. The results indicate that our model can predict sufficiently both the reviews of our dataset and the reviews that we created.

  Multiple approaches could tackle the task of Sentiment Analysis. As we know machine learning models and neural networks don't take raw text data as an input. This means we must somehow encode our textual data to numeric values that our models can understand. There are many different ways of doing this like using the Bag of Words method. This is a pretty easy technique where each word in a sentence is encoded with an integer and thrown into a collection that does not maintain the order of the words but does keep track of the frequency. Another way of doing it or integer encoding. This involves representing each word or character in a sentence as a unique integer and maintaining the order of these words. We could even use GloVe Embeddings,which are a type of word embedding that encode the co-occurrence probability ratio between two words as vector differences. GloVe uses a weighted least squares objective that minimizes the difference between the dot product of the vectors of two words and the logarithm of their number of co-occurrences.

  Additionaly, there are plenty of ways to build our Neural Network, there is a large number of bibliography that suggest multiple architectures on how to build the desire Neural Model. Moreover, another possible way of facing this problem would be to use a test set different from our training set, so, for example, we could have used the famous IMDB dataset in order to build our model and then we could implement our model having as a test set the Tweets dataset. In this way, we could have built an even stronger model that predicts out of the box sentences.

References

I. Chowdhary, KR1442. "Natural language processing." Fundamentals of artificial intelligence (2020): 603-649. Medhat, Walaa, Ahmed Hassan, and Hoda Korashy. "Sentiment analysis algorithms and applications: A survey." Ain Shams engineering journal 5.4 (2014): 1093-1113.

II. Medhat, Walaa, Ahmed Hassan, and Hoda Korashy. "Sentiment analysis algorithms and applications: A survey." Ain Shams engineering journal 5.4 (2014): 1093-1113.

III. Feldman, Ronen. "Techniques and applications for sentiment analysis." Communications of the ACM 56.4 (2013): 82-89.

IV. Vijayarani, S., Ms J. Ilamathi, and Ms Nithya. "Preprocessing techniques for text mining-an overview." International Journal of Computer Science Communication Networks 5.1 (2015): 7-16

V. Kannan, Subbu, et al. "Preprocessing techniques for text mining." International Journal of Computer Science Communication Networks 5.1 (2014): 7-16.

VI. Alzahrani, Sultan, et al. "A network-based model for predicting hashtag breakouts in Twitter." International Conference on Social Computing, Behavioral-Cultural Modeling, and Prediction. Springer, Cham, 2015.

VII. Cheng, Henry Jen-Hao. Empirical Study on the Effect of Zero-Padding in Text Classification with CNN. University of California, Los Angeles, 2020

VIII. Bogale Gereme, Fantahun, and William Zhu. "Fighting fake news using deep learning: Pre-trained word embeddings and the embedding layer investigated." 2020 The 3rd International Conference on Computational Intelligence and Intelligent Systems. 2020.