**CE314 - Text Classification on IMDB dataset**

Andrei Alexandru Talpan

[**Link to Google colab notebook**](https://colab.research.google.com/drive/1KrmL9B8kbssRlhGgjA53Y-8P6upaFD2C?usp=sharing)

**Abstract**

This report presents a simple and efficient neural language model for text classification, a topic of increasing importance in business contexts as it enables the analysis of customer data such as product or service reviews. The proposed model utilizes a neural network architecture to classify reviews as positive or negative, with training and testing performed on a dataset of 50000 labelled reviews from the Internet Movie Database (IMDb). The report begins with an introduction to natural language processing (NLP) in Section I. The background of sentiment analysis is presented in Section II, followed by a description of the proposed solution in Section III. The results of the model are discussed in Section IV, and the report concludes with a summary of the proposed solution in Section V.

***Section I: Introduction***  
  
As tech and especially AI sector evolves, applications of Natural Language Processing are becoming a prevalent aspect in our lives.

According to a report by [Markets and Markets](https://www.marketsandmarkets.com/Market-Reports/natural-language-processing-nlp-825.html), “The global Natural Language Processing (NLP) market size to grow from USD  11.6 billion in 2020 to USD 35.1 billion by 2026, at a Compound Annual Growth Rate (CAGR) of 20.3% during the forecast period.” The report also mentions that, “The rise in the adoption of NLP-based applications across verticals to enhance customer experience and increase in investments in the healthcare vertical is expected to offer opportunities for NLP vendors.” Currently, there are **many applications of NLP** in our daily live from machine translations (ex: Google translate), sentiment analysis (Social media platforms filtering inappropriate content) to text to speech (Apple’s Siri, Amazon Alexa) and document classification (AWS Comprehend service). Advanced NLP techniques are now capable of assessing real-time sentiment, content toxicity, intelligent chatbots, and hot / trending topics occurring on platforms such as Twitter, Facebook, Google search. Without any doubts, this emerging use of AI, can provide companies with competitive advantages in the market.

However, there are also restraints and concerns over the massive use of NLP apps. The technology behind uses neural networks and deep learning techniques for all sorts of data: text, time series, financial data, speech, audio and video.

Some potential security and privacy concerns with NLP models include:

1. **Data privacy**: NLP models often require access to large amounts of sensitive text data, such as private messages or medical records. There is a risk that this data could be accessed or misused by unauthorized parties, leading to privacy violations.
2. **Bias**: NLP models can sometimes exhibit bias in their predictions, based on the data they have been trained on. This can lead to unfair or discriminatory outcomes, especially if the data is biased in some way.

***Section II: Sentiment Analysis background***

Sentiment analysis is a process of extracting information about a subject and identifying its characteristics. The objective is to decide whether a text contains positive, negative or neutral views. Currently, there are three approaches to address this problem of sentiment analysis [1]: lexicon-based techniques, machine-learning-based techniques, and hybrid approaches.

**Lexicon-based techniques** were the first ones to be used, having two approaches: dictionary-based and corpus-based [2]. In the first case, classification is performed using a dictionary of terms, like WordNet. On the other hand, the corpus-based method uses statistical analysis of contents of documents collections combined with techniques such as hidden Markov models(HMM) [3].

**Machine-learning-based** techniques [4] proposed are formed of two groups: traditional models and deep learning models. First models, refer to classifiers such as such as the naïve Bayes classifier [5], maximum entropy classifier [6, 7], or support vector machines (SVM) [8]. Inputs to these being lexical features, parts of speech, feature matrices etc. Deep learning models achieve better results as they use complex neural networks architectures such as CNN, DNN and RNN consider multiple parameters.

The **hybrid** approaches [9] combine the previously mentioned categories of models.

Regardless of the approach (deep learning or traditional ML ), the sentiment classification task requires clean text training data. meaning removing punctuation marks, non-characters and stop words.

After pre-processing the data, the texts can be split into individual words, these can be further transformed using lemmatization or stemming.

As machines can only understand numbers, we must do text vectorization, transforming words to numbers. This is done by converting words into numerical vectors by using word embeddings or term frequency-inverse document frequency (TF-IDF).

Word embedding [4] is a technique for language modelling and feature learning in which words are mapped to a vector of real values in a way that preserves semantic relationships between words. The goal of term frequency-inverse document frequency (TF-IDF) is to determine the importance of a word in a collection of documents or corpus. This metric is often used in keyword extraction and information retrieval and is a commonly employed method of vectorization in natural language processing (NLP). Both word embedding and TF-IDF have been utilized as input features for deep learning algorithms in NLP.

***Section III Proposed solution***

In the initial phase of the project, the data import and exploration process involved reading the CSV file and determining the shape of the dataset. It was necessary to encode the "positive" and "negative" labels as numerical values of 0 and 1, respectively. The dataset was then split into 80% training data (40000 records) and 20% testing data (10000 records). Preprocessing of the training data was conducted, including the removal of HTML tags, punctuation marks, stop words, non-characters. After the data had been cleaned, it needed to be prepared for input into the model. This process included the conversion of the text data into a numerical data structure suitable for machine learning through the use of text tokenization, in which texts are split into individual tokens (words) and assigned a unique integer index. For this task, the built-in Tokenizer class from Keras was utilized, including the creation of a dictionary for the entire dataset (mapping each word to a unique integer index) and the conversion of words in each review text into integer sequences. However, the resulting sequences were of varying lengths, which posed a problem for the deep learning model due to its need for uniform-sized input. To address this issue, padding was applied, meaning that all sequences were extended to a specific length (in this case, the maximum length of all reviews from the training set).

The model’s structure is composed of three sequential layers:

* **Embedding** layer  
  This converts the indexes associated with words to dense vectors of fixed size, and places them into an embedding matrix.

These dense vectors are called "embeddings". They can be learned by the neural network during training, and they allow the network to understand the relationships or similarity between different words.  
[10]Table

Description automatically generated  
here, the **vectors get updated during training.** For example, the word “deep” gets represented by a vector [.65, .21, .25, .45, .78, .82]. 

* **GlobalAveragePooling1D**

Its input is a sequence of words or characters and the output is a sequence of features representing the input. It works by taking the average of all the values in the input along the specified dimension.

* **Dense** connected layer with 1 output  
  It receives input from all neurons of its previous layer. Uses a sigmoid activation that because it produces outputs that can be interpreted as probabilities. For example, if the output of a sigmoid activation function is 0.7, it can be interpreted as a 70% probability that the input belongs to a certain class.

See the model’s structure below:

Timeline

Description automatically generated

Since this is a binary classification problem and the model outputs a probability, the **loss function** used is BinaryCrossentropy.

Finally, we make predictions on the test dataset and get predictions in an array format as the one below: (5 predictions for 5 distinct movies)

*[[0.12902652], [0.69782865], [0.5216292], [0.92247415], [0.05916589] ]*

It can be observed that the closer the value is to 0, the more "positive" the text content is, while values closer to 1 indicate a greater degree of "negativity.

Example:

[0.12902652] ------- positive

----------

If you like original gut wrenching laughter you will like this movie. If you are young or old then you will love this movie, hell even my mom liked it.<br /><br />Great Camp!!!

The solution was developed using Keras API (high-level library built on top of Tensorflow) and model training was performed in a Google Colab environment using GPU acceleration for faster computation.

***Section IV Results***

The latest execution achieved an accuracy of 88% and a loss of 0.30. This high performance can be attributed to several factors, including the use of a large and qualitatively high number of movie review samples, the application of data cleaning techniques, and the balance of negative and positive samples in the dataset. The accuracy and loss during training are displayed in the accompanying plot, along with a confusion matrix.A picture containing shape

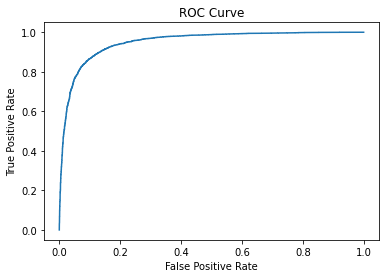
Description automatically generated

Application

Description automatically generated with medium confidence

Another important metric is the **ROC AUC** score.

The area under the curve (AUC) is a measure of how well a classifier can distinguish between positive and negative classes. The TPR is the ratio of true positive predictions to the total number of positive instances in the dataset. The FPR is the ratio of false positive predictions to the total number of negative instances in the dataset.



model AUC score: 0.9497853

The AUC is a measure of how well a classifier can distinguish between positive and negative classes and can range from 0 to 1, with a value of 1 indicating perfect discrimination.

***Section V Conclusion***

The presented solution proves modern NLP libraries have greatly simplified the process of building accurate models, by providing high-level APIs for development that reduces many of the low-level complexities.

The current model can be further improved by several techniques such as **using larger and more diverse dataset**, **fine-tuning hyperparameters, using pre-trained models and transfer learning, incorporating domain knowledge and external sources.**

Overall, deep learning is a powerful but complex field that requires a strong understanding of machine learning concepts and a willingness to experiment and iterate to achieve good results.

The proposed solution demonstrates a relatively high level of accuracy, making it suitable for use in non-critical practical settings. While the model's accuracy is not quite at the level of 99.99%, it could still be applied in closed environments such as schools, universities, and small businesses to evaluate customer feedback of a service or app. In these contexts, the model could serve as an initial indicator of customer experience, which could be further enhanced by incorporating other indicators such as star ratings, active user counts, purchase volumes, and pricing.

**References**

1. Bhavitha, B.; Rodrigues, A.P.; Chiplunkar, N.N. Comparative study of machine learning techniques in sentimental analysis. In Proceedings of the 2017 International Conference on Inventive Communication and Computational Technologies (ICICCT), Coimbatore, India, 10–11 March 2017; pp. 216–221.
2. Soni, S.; Sharaff, A. Sentiment analysis of customer reviews based on hidden markov model. In Proceedings of the 2015 International Conference on Advanced Research in Computer Science Engineering & Technology (ICARCSET 2015), Unnao, India, 6 March 2015; pp. 1–5.
3. Soni, S.; Sharaff, A. Sentiment analysis of customer reviews based on hidden markov model. In Proceedings of the 2015 International Conference on Advanced Research in Computer Science Engineering & Technology (ICARCSET 2015), Unnao, India, 6 March 2015; pp. 1–5.
4. Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.S.; Dean, J. Distributed representations of words and phrases and their compositionality. In Proceedings of the Advances in neural Information Processing Systems, Lake Tahoe, NV, USA, 5–10 December 2013; pp. 3111–3119.
5. Malik, V.; Kumar, A. Communication. Sentiment Analysis of Twitter Data Using Naive Bayes Algorithm. *Int. J. Recent Innov. Trends Comput. Commun.* **2018**, *6*, 120–125.
6. Mehra, N.; Khandelwal, S.; Patel, P. *Sentiment Identification Using Maximum Entropy Analysis of Movie Reviews*; Stanford University: Stanford, CA, USA, 2002.
7. Wu, H.; Li, J.; Xie, J. Maximum entropy-based sentiment analysis of online product reviews in Chinese. In *Automotive, Mechanical and Electrical Engineering*; CRC Press: Boca Raton, FL, USA, 2017; pp. 559–562.
8. Firmino Alves, A.L.; Baptista, C.d.S.; Firmino, A.A.; Oliveira, M.G.d.; Paiva, A.C.D. A Comparison of SVM versus naive-bayes techniques for sentiment analysis in tweets: A case study with the 2013 FIFA confederations cup. In Proceedings of the 20th Brazilian Symposium on Multimedia and the Web, João Pessoa, Brazil, 18–21 November 2014; pp. 123–130.
9. Medhat, W.; Hassan, A.; Korashy, H. Sentiment analysis algorithms and applications: A survey. *Ain Shams Eng. J.* **2014**, *5*, 1093–1113.

10 [Deep Learning #4: Why You Need to Start Using Embedding Layers | by Rutger Ruizendaal | Towards Data Science](https://towardsdatascience.com/deep-learning-4-embedding-layers-f9a02d55ac12)