## 

SENTIMENTAL ANALYSIS

05.28.2022

**─**

NGUEKENG TIEUMENA Marc Donald

National Advanced School of Engineering of Yaounde

(+237)699844274

marcdonald521[@gmail.com](mailto:julienmingo@gmail.com)

4th year computer engineering

# The Challenge

The goal of this challenge is to build and compare different approaches which ideally improve upon the provided baselines.

# Objectives

1. Present the other approaches.
2. Precise how the training/validation and test set have been developed.
3. Present different models used and observed results.
4. Present the strengths and weaknesses of models used
5. Choose metric(s) and justify their choice .

# Dataset

The dataset which the challenge requires us to analyse is the IMDB Large Movie Review Dataset. It contains 50 000 (fifty thousand) movie reviews made of text representing the viewer’s comments along with a label “positive” or “negative”.

Depending on the approach, we used either the dataset **IMDB Dataset.csv** arranged in tables, or the **Large Movie Review Dataset v1.0** which is structured in directories.

# The approaches

Text preprocessing is traditionally an important step for natural language processing (NLP) tasks. It transforms text into a more digestible form so that machine learning algorithms can perform better on it. There are a multitude of approaches to resolve NLP problems. They include the following:

## Words embedding

The provided baseline uses bags of words approach and TF-IDF approach. But, a potential drawback with these approaches is that the feature vector for each document can be huge. For instance, if we have a half million unique words in our corpus and we want to represent a sentence that contains 10 (ten) words, our feature vector will be a half million dimensional one-hot encoded vector where only 10 indexes will have 1(one). This is a waste of space and it increases the algorithm’s complexity exponentially resulting in the curse of dimensionality. In word embeddings, every word is represented as an n-dimensional dense vector. The vector size is small and none of the indices in the vector is actually empty.

In our notebook we used the tokenizer class of keras. This tokenizer class allows us to choose which words will be represented. So only those with particular frequencies will be used as dimensions of our vector space.

We used ,for this approach, **IMDB Dataset.csv.**

## BERT (Bidirectional Encoder Representations from Transformers)

As said earlier, we can find more economic approaches. BERT is one of them. The provided baseline uses several machine learning algorithms (logistic regression, support vector machine, naive bayes), which do not give very good results as well as the preprocessing method used. So, in this approach, in addition to the bert tokenization, we used the pre-trained model BERT.

# About the split

We used a train set, a validation set and a test set randomly. The test set represents 20% of the dataset. We took 20% of the rest of the dataset as the validation set and the remaining 64% of the dataset as the training set.

# 

# 

# Models

## LSTM

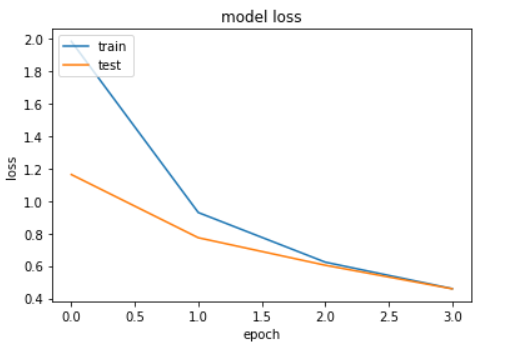
Long Short-Term Memory (LSTM) networks are a type of recurrent neural network capable of learning order dependence in sequence prediction problems. So it is suitable for our sentiment analysis problem since the data at our disposal is sequential.

**Strength:** LSTM is well-suited to classify, process and predict time series given time lags of unknown duration. Relative insensitivity to gap length gives an advantage to LSTM over alternative RNNs, hidden Markov models and other sequence learning methods.

**Weakness:** In short, LSTM requires 4 linear layers (MLP layer) per cell to run at and for each sequence time-step. Linear layers require large amounts of memory bandwidth to be computed, in fact they cannot use many compute units often because the system has not enough memory bandwidth to feed the computational units.

In our notebook we used this model with the word embedding approach.

Observed results :



On the test set we obtained : loss: 0.4564 - accuracy: 0.8765

## CNN

CNNs are a class of Deep Neural Networks that can recognize and classify particular features from images and are widely used for analyzing visual images. Their applications range from image and video recognition, image classification, medical image analysis, computer vision and natural language processing.

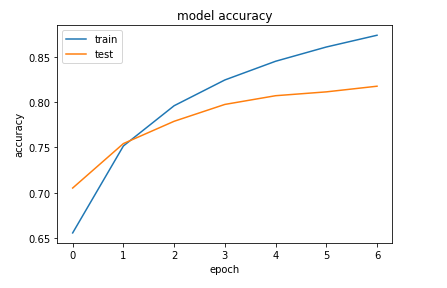
Convolutional neural networks have been found to work well with text data as well. Though text data is one-dimensional, we can use 1D convolutional neural networks to extract features from our data.

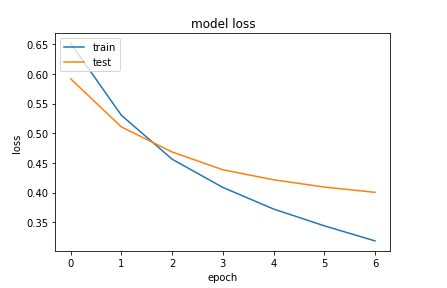
**Strength**: CNNs are faster than RNN and LSTM.

**Weakness**: CNNs aren’t capable of learning order dependence in sequence prediction problems.

We used this model with word embedding too

Observed results :





On the test set we obtained : loss: 0.3820 - accuracy: 0.8251

## Pre-trained model BERT with ktrain

Ktrain is a lightweight wrapper for the deep learning library TensorFlow Keras (and other libraries) used to help build, train, and deploy neural networks and other machine learning models.

We used ktrain to efficiently build the BERT approach

**Strength**: BERT analyzes every sentence with no specific direction, it does a better job at understanding the meaning of homonyms than previous NLP methodologies, such as embedding methods.

**Weakness**: The main drawback of using BERT is the computational resources needed to train/fine-tune and make inferences.

Here we obtained 0.94 as accuracy.

# Metric(s)

Evaluating a model is a core part of building an effective machine learning model. The metric we used in our work is binary accuracy, since we are required to perform a binary classification. As the target variable is not continuous, the binary classification model predicts the probability of a target variable to be either positive or negative.