NATIONAL INSTITUTE OF TECHNOLOGY CALICUT

EE2026D

ARTIFICIAL NEURAL NETWORKS AND FUZZY LOGIC SYSTEMS

COURSE PROJECT: Sentiment Analysis Based on News Headline

Participants and Contribution:

- Krishna Agrawal (B200313EE) -50%
- Dinesh Kumar (B1900313EE) -50%

1. Abstract

In this project, we try to implement a News Headline sentiment analysis model that helps to overcome the challenges of identifying the sentiments of the news headlines. We created sentiment predictions using Natural Language Processing (NLP) approaches. News-Headlines-Dataset-For-Sarcasm-Detection was used for testing and training of the model. We have used popular python libraries such as TensorFlow, Keras and Machine learning techniques such as Adam Optimizer to train the model. It was successful in predicting the correct headline on a new given instance with an accuracy of 82.89%.

2. Approach

In this analysis we identify ether a news headline is sarcastic or not sarcastic by training a neural network using supervised learning approach. Training dataset contain sentences which are classified into 1(sarcastic) and 0(non-sarcastic). The output column will range from 0 to 1. If the output is greater than 0.5 then it is sarcastic, if less than 0.5 then it is not sarcastic.

3. Data

The dataset was taken from github repository (https://github.com/rishabhmisra/News-Headlines-Dataset-For-Sarcasm-Detection).

Each record consists of three attributes:

- is_sarcastic: 1 if the record is sarcastic otherwise 0
- headline: the headline of the news article
- article_link: link to the original news article. Useful in collecting supplementary data

We will only be using is_sarcastic (as labels in code) and headline (as sentences in code) to create this model. The data is divided into testing and training data. First 20,000 headlines will be used to train the modal and the next 8,619 headline for testing the model.

4. Background Theory

4.1 Natural Language Processing (NLP)

Natural Language Processing (NLP) is a tract of Artificial Intelligence, devoted to make computers understand the statements or words written in human languages. We can define language as a set of rules or set of symbol. For conveying information broadcasting it we can combine these symbols and use them. Symbols are combined using a set of rules. Natural Language Processing basically can be classified into two parts i.e. Natural Language Understanding and Natural Language Generation which evolves the task to understand and generate the text. The metric of NLP assess on an algorithmic system allows for the integration of language understanding and language generation.

4.2. Tokenization

Tokenization is the very first step in most text processing works. Tokenization is the process of breaking up a string into tokens. Commonly, these tokens are words, numbers, and/or punctuation. We can simply split text by whitespaces for this project. Every word which we get after splitting is then assigned a token (number). Note that punctuation marks are ignored and not converted into tokens.

4.3. Activation Function That Are Used

There are two activation function used in this project Sigmoid Function and ReLU Function.

4.3.1 Sigmoid Function

This is a smooth function and is continuously differentiable. This is one of the most used non-linear activation functions. It is mostly used at the output layer and it is a widely used activation function for the binary classification and our model is a binary classification model. The function ranges between 0-1.

$$\frac{1}{(1+e^{-x})}$$

4.3.2 ReLU Function

The ReLU activation function is a non-linear activation function stands for Rectified Linear Unit. ReLU function is a simple and robust when compared to other activation function.

It is represented by:

$f(x) = \max(0,x)$ $(-\inf < x < +\inf)$

4.4. Cost Function Optimizers

Our goal during the ANN training process is to minimize the amount of error we are making, the difference between the predicted value and the real value of the output. So, in order to learn about these different signals is to predict the output value of then compute the cost associated with this iteration or epoch, determine which set of weights at this stage will minimize the cost function, and use this to update the weights. We repeat this process as many times to minimize the cost function.

Here we use Adam optimizer to minimize the cost function by determining the weights.

4.4.1 Adam optimizer

Adaptive Moment Estimation is an algorithm for optimization technique for gradient descent. The method is really efficient when working with large problem involving a lot of data or parameters. It requires less memory and is efficient.

4.5. Binary_crossentropy

The purpose of loss functions is to compute the quantity that a model should seek to minimize during training. It is used as a loss function for binary classification model. The binary_crossentropy function computes the cross-entropy loss between true labels and predicted labels.

4.6. EMBEDDING

The word embedding is a class of techniques where individual words are represented as real-valued vectors in a predefined vector space. Each word is given a vector and the values of the vector are updated according to its occurrence in the sarcastic and non- sarcastic sentences in the training dataset.

5. DATA PREPROCESSING

5.1. Data Splitting

We first split the original data into two parts training dataset and testing dataset. The original data set has 28,619 headlines. First 20,000 will be used for training the model and the next 8,619 will be used for testing.

5.2. Tokenization and Sequencing

In this process we break down all the sentences in the training into words and take the 1000 most frequent and assignee them a number.

Then we convert every headline in training and testing dataset to sequence of number in which every number represents the tokenized value of the word in a headline. If a word which is not present in the tokenized word list will be assigned number 1.

Now to make these sequences ready for testing and training we will have to make all of them of same size. So we convert every sequence to a size of 20. Every blank space after the sequence will be represented by zero and if the list sequence is bigger then 20, then every tokenized value after which comes after that will be ignored.

6. NEURAL NETWORK MODEL

- The top layer is an embedding where the direction (vector) of each word will be learned epoch by epoch.
- Then we pool with a global average pooling i.e. adding up the vectors.
- This is then fed into a common deep neural network.
- First dense layer has an activation function of ReLU (Rectified Linear Unit).
- The output dense layer has an activation function of Sigmoid Function.

```
Model: "sequential_18"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_18 (Embedding)    (None, 20, 16)            16000

 global_average_pooling1d_14  (None, 16)               0
 (GlobalAveragePooling1D)

 dense_36 (Dense)            (None, 24)                408

 dense_37 (Dense)            (None, 1)                 25

=================================================================
Total params: 16,433
Trainable params: 16,433
Non-trainable params: 0
_________________________________________________________________
```

7. RESULT

Now we train our model using training data and then test it using testing data. For each epoch the accuracy and loss of training and testing data are shown.

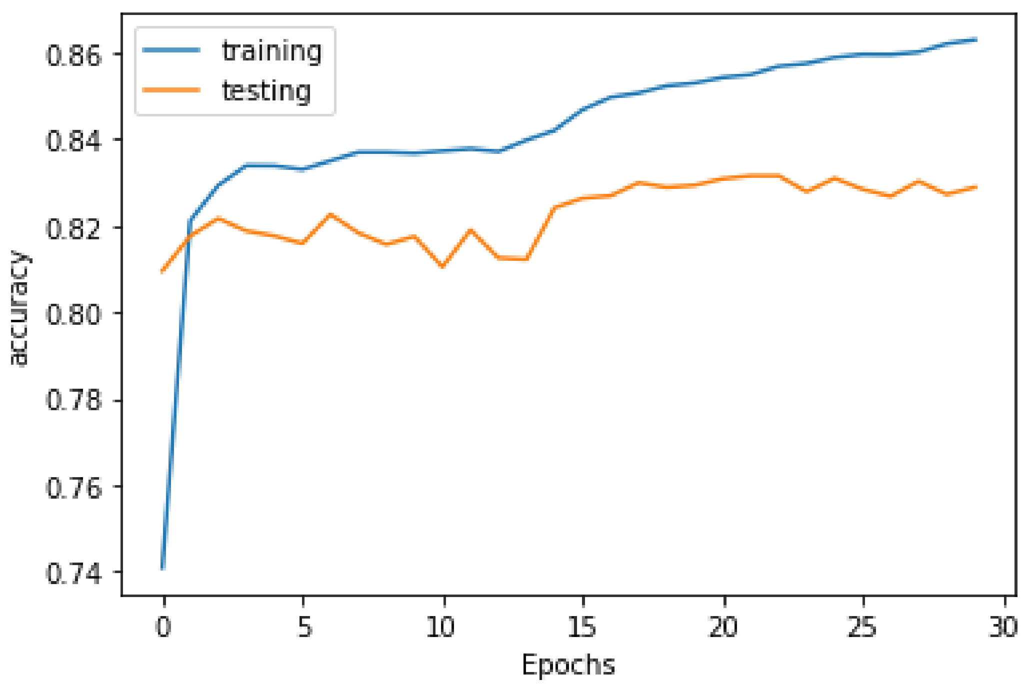
Accuracy after 30 epochs for training data is 86.30% and for testing data 82.89%.

```
Epoch 1/30
625/625 - 2s - loss: 0.5413 - accuracy: 0.7408 - val_loss: 0.4094 - val_accuracy: 0.8095 - 2s/epoch - 4ms/step
Epoch 2/30
625/625 - 2s - loss: 0.3901 - accuracy: 0.8212 - val_loss: 0.3875 - val_accuracy: 0.8176 - 2s/epoch - 3ms/step
Epoch 3/30
625/625 - 2s - loss: 0.3704 - accuracy: 0.8293 - val_loss: 0.3865 - val_accuracy: 0.8217 - 2s/epoch - 3ms/step
Epoch 4/30
625/625 - 2s - loss: 0.3646 - accuracy: 0.8339 - val_loss: 0.3883 - val_accuracy: 0.8188 - 2s/epoch - 3ms/step
Epoch 5/30
625/625 - 2s - loss: 0.3612 - accuracy: 0.8339 - val_loss: 0.3897 - val_accuracy: 0.8176 - 2s/epoch - 2ms/step
Epoch 6/30
625/625 - 2s - loss: 0.3603 - accuracy: 0.8330 - val_loss: 0.3933 - val_accuracy: 0.8159 - 2s/epoch - 3ms/step
```

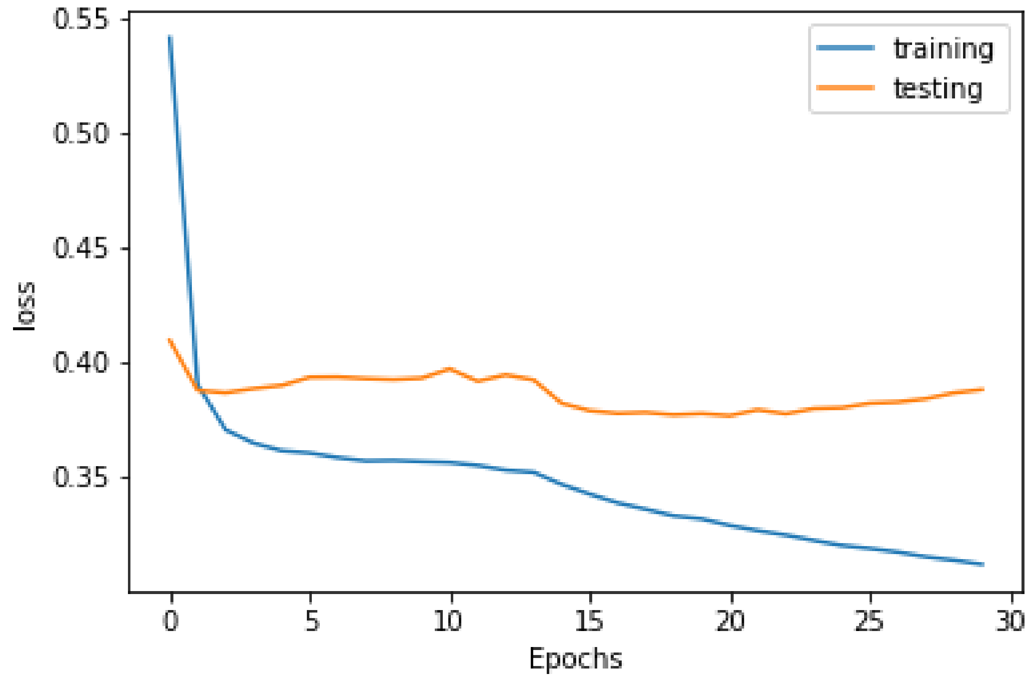
```
Epoch 26/30
625/625 - 2s - loss: 0.3186 - accuracy: 0.8597 - val_loss: 0.3819 - val_accuracy: 0.8284 - 2s/epoch - 4ms/step
Epoch 27/30
625/625 - 2s - loss: 0.3171 - accuracy: 0.8596 - val_loss: 0.3824 - val_accuracy: 0.8268 - 2s/epoch - 3ms/step
Epoch 28/30
625/625 - 2s - loss: 0.3150 - accuracy: 0.8602 - val_loss: 0.3839 - val_accuracy: 0.8303 - 2s/epoch - 3ms/step
Epoch 29/30
625/625 - 2s - loss: 0.3135 - accuracy: 0.8621 - val_loss: 0.3864 - val_accuracy: 0.8272 - 2s/epoch - 3ms/step
Epoch 30/30
625/625 - 2s - loss: 0.3117 - accuracy: 0.8630 - val_loss: 0.3879 - val_accuracy: 0.8289 - 2s/epoch - 3ms/step
```

7.1. Training Curves

- **Model Accuracy Curve-**



- Model loss Curve



8. CONCLUSIONS

In this paper we have created a Sentiment Analysis Based on News Headline model. For testing our model we use two new sentences one sarcastic and one non- sarcastic.

"reports of movie being good reach area man"

"behind the scenes of an intricate fbi sting"

```
1/1 [==============================] - 0s 73ms/step
[[0.99936676]
 [0.04564513]]
```

For the first sentences the model gives us 0.99 value which indicates highly sarcastic. For the second sentence value is 0.04 which indicates non- sarcastic.

9. REFERENCES

- https://www.analyticsvidhya.com/blog/2022/08/theory-behind-the-basics-of-nlp/
- https://www.tokenex.com/blog/ab-what-is-nlp-natural-language-processing-tokenization/
- https://medium.com/sfu-cspmp/nlp-word-embedding-techniques-for-text-analysis-ec4e91bb886f
- https://github.com/rishabhmisra/News-Headlines-Dataset-For-Sarcasm-Detection