Journal of Ubiquitous Computing and Communication Technologies (UCCT) (2021)
Vol.03/ No.02
Pages: 61-79
https://www.irojournals.com/jucct/
DOI: https://doi.org/10.36548/jucct.2021.2.001

Maximizing the Prediction Accuracy in Tweet Sentiment Extraction using Tensor Flow based Deep Neural Networks

Thivaharan Sakthivadivel[1], Srivatsun Gopalakrishnan[2]
[1]Department of Computer Science and Engineering, PSG Institute of Technology and Applied Research, Coimbatore, India.
[2]Department of Electronics and Communication Engineering, PSG College of Technology, Coimbatore, India.

Abstract: The amount of data generated by modern communication devices is enormous, reaching petabytes. The rate of data generation is also increasing at an unprecedented rate. Though modern technology supports storage in massive amounts, the industry is reluctant in retaining the data, which includes the following characteristics: redundancy in data, unformatted records with outdated information, data that misleads the prediction and data with no impact on the class prediction. Out of all of this data, social media plays a significant role in data generation. As compared to other data generators, the ratio at which the social media generates the data is comparatively higher. Industry and governments are both worried about the circulation of mischievous or malcontents, as they are extremely susceptible and are used by criminals. So it is high time to develop a model to classify the social media contents as fair and unfair. The developed model should have higher accuracy in predicting the class of contents. In this article, tensor flow based deep neural networks are deployed with a fixed Epoch count of 15, in order to attain 25% more accuracy over the other existing models. Activation methods like "Relu" and "Sigmoid", which are specific for Tensor flow platforms support to attain the improved prediction accuracy.

Keywords: Corpus, Data analytics life cycle, kaggle, Tensorflow, Tuples, Vectors, Keras, RELU, Sigmoid, ADAM and Cross Entropy.

1. INTRODUCTION – PROBLEM STATEMENT

ISSN: 2582-337X
Submitted:08.06.2021
Revised: 09.06.2021
Accepted: 10.06.2021
Published: 14.06.2021

Journal of Ubiquitous Computing and Communication Technologies (UCCT) (2021)
Vol.03/ No.02
Pages: 61-79
https://www.irojournals.com/jucct/
DOI: https://doi.org/10.36548/jucct.2021.2.001

This article proposes a method for analyzing the tweet contents in order to identify the malcontents. For this purpose, a corpus [1] from "kaggle repository" is used. The completion of overall process involves various stages and data modifiers, which accounts to the development of a neural network model. A statistical study states that, the connections among the stages and modifiers (i.e. layers) are sometimes tightly coupled and majority of the time it will be loosely coupled. This article aims to create a model to minimize the loss in prediction accuracy through various permutation along with a combination of underlying neural network layers.

The main problems that exist in the current techniques are: inability to perform well in different domains, inadequate accuracy and performance in sentiment analysis based on insufficient labeled data, incapability to deal with complex sentences that require more sentiment words and simple analysis. Opinion mining and sentiment analysis technology necessitates the completion of several tasks, each of which poses substantial challenges. In general, this field is still in its nascent stage. Majority of tools still struggle to determine what truly constitutes a negative, neutral, or positive expression. Also, there is no proper evidence to predict the applicability about the mechanics behind it but at the moment it is not advanced enough to successfully deal with sarcasm or context of the sentences.

1.1 LIFE CYCLE OF DATA ANALYTICS

1.2 DATASET

The overall construction of this article follows the standards and stages cited in the data analytics life cycle [2]. A dataset from the "Kaggle repository [3]" is used. The dataset has a total of 30186 tuples and out of which, 26752 tuples (89% of total data) are used as the training set and 3434 tuples (11% of the data) are used as the test set. The dataset is validated for the possible classes of "Positive", "negative" and "neutral". The dataset has the following columns: "textID", "text", "selected_text" and "sentiment". From these, the "selected_text" column is not considered for the prediction as it is just a replica of the "text" column and is given less weightage. The figure 1(a) and 1(b) shows the view of the original dataset under consideration and the dataset with the selected column respectively.

ISSN: 2582-337X
Submitted:08.06.2021
Revised: 09.06.2021
Accepted: 10.06.2021
Published: 14.06.2021

Journal of Ubiquitous Computing and Communication Technologies (UCCT) (2021)
Vol.03/ No.02
Pages: 61-79
https://www.irojournals.com/jucct/
DOI: https://doi.org/10.36548/jucct.2021.2.001

	A	B	C	D
1	textID	text	selected_text	sentiment
2	cb774db0d1	I`d have responded, if I were going	I`d have responded, if I were going	neutral
3	549e992a42	Sooo SAD I will miss you here in San Diego!	Sooo SAD	negative
4	088c60f138	my boss is bullying me...	bullying me	negative
5	9642c003ef	what interview! leave me alone	leave me alone	negative
6	358bd9e861	Sons of ****, why couldn`t they put them c	Sons of ****,	negative
7	28b57f3990	http://www.dotheboucny.com/smf - some	http://www.dotheboucny.com/smf - some sh	neutral
8	6e0c6d75b1	2am feedings for the baby are fun when he	fun	positive
9	50e14c0bb8	Soooo high	Soooo high	neutral
10	e050245fbd	Both of you	Both of you	neutral
11	fc2cbefa9d	Journey!? Wow... u just became cooler. he	Wow... u just became cooler.	positive
12	2339a9b08b	as much as i love to be hopeful, i reckon th	as much as i love to be hopeful, i reckon the c	neutral
13	16fab9f95b	I really really like the song Love Story by Tay	like	positive
14	74a76f6e0a	My Sharpie is running DANGERously low on	DANGERously	negative
15	04dd1d2e34	i want to go to music tonight but i lost my v	lost	negative
16	bbe3cbf620	test test from the LG enV2	test test from the LG enV2	neutral

Figure 1(a). The original Kaggle dataset

	A	B	C
1	textID	text	sentiment
2	f87dea47db	Last session of the day http://twitpic.com/67ezh	neutral
3	96d74cb729	Shanghai is also really exciting (precisely -- skyscrapers galore). Good tweeps in	positive
4	eee518ae67	Recession hit Veronique Branquinho, she has to quit her company, such a shame	negative
5	01082688c6	happy bday!	positive
6	33987a8ee5	http://twitpic.com/4w75p - I like it!!	positive
7	726e501993	that`s great!! weee!! visitors!	positive
8	261932614e	I THINK EVERYONE HATES ME ON HERE lol	negative
9	afa11da83f	soooooo wish i could, but im in school and myspace is completely blocked	negative
10	e64208b4ef	and within a short time of the last clue all of them	neutral
11	37bcad24ca	What did you get? My day is alright.. haven`t done anything yet. leaving soon to	neutral
12	24c92644a4	My bike was put on hold...should have known that.... argh total bummer	negative
13	43b390b336	I checked. We didn`t win	neutral
14	69d6b5d93e	.. and you`re on twitter! Did the tavern bore you that much?	neutral
15	5c1e0b61a1	I`m in VA for the weekend, my youngest son turns 2 tomorrow.....it makes me kir	negative

Figure 1(b). The dataset with the impact columns

1.3 THE ENVIRONMENT FOR THE MODEL

The proposed framework incorporates the Tensorflow [4], which is a machine learning based end-to-end synchronous and open source platform. Tensorflow has an ever increasing updated set of libraries and community feeds, which are available in the form of resources. The Tensorflow 2.1 version is used here and the entire experiment is studied by using an AMD GPU (Graphical Processing Unit) machine with PNY GeForce 1660 / 6GB graphics card.

The flow of tweet sentiment prediction is outlined in the Figure 2.

ISSN: 2582-337X
Submitted:08.06.2021
Revised: 09.06.2021
Accepted: 10.06.2021
Published: 14.06.2021

Journal of Ubiquitous Computing and Communication Technologies (UCCT) (2021)
Vol.03/ No.02
Pages: 61-79
https://www.irojournals.com/jucct/
DOI: https://doi.org/10.36548/jucct.2021.2.001

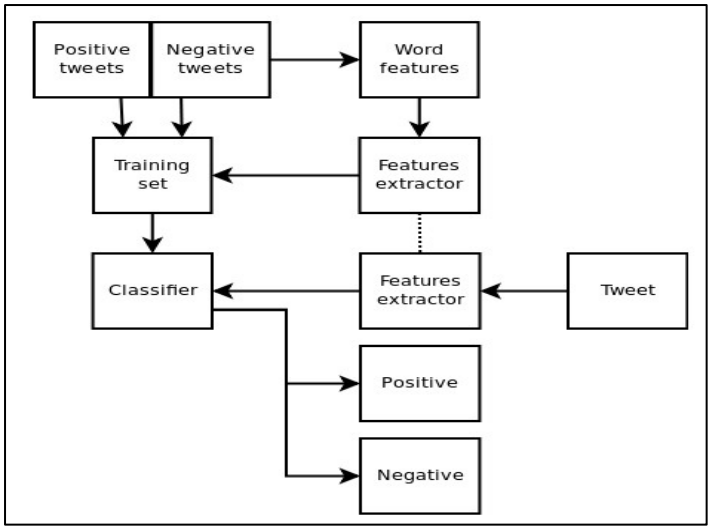

Figure 2. Flow of Tweet sentiment extraction

2. DATA PREPROCESSING

As the input dataset contains elements (like redundant spaces (white spaces), expressive punctuation marks and emojis, unwanted URL reference links) which need to be eliminated, a primary data preprocessing has been done.

2.1 TENSORFLOW PACKAGES FOR THE SETUP

The following packages and modules are imported to preprocess the data: tensorflow, Tensorflow.keras, sklearn.preprocessing and sklearn.model_selection. Along with these packages, the modules like layers, losses and regularizers are also imported. For data oriented manipulation, the numpy and pandas are also imported. The code for the same is shown below:

```
import Tensorflow as TF
from tensorflow.keras import layers
from tensorflow.keras import losses
from tensorflow.keras import regularizers
import pandas as PD
```


ISSN: 2582-337X
Submitted:08.06.2021
Revised: 09.06.2021
Accepted: 10.06.2021
Published: 14.06.2021

Journal of Ubiquitous Computing and Communication Technologies (UCCT) (2021)
Vol.03/ No.02
Pages: 61-79
https://www.irojournals.com/jucct/
DOI: https://doi.org/10.36548/jucct.2021.2.001

```
import numpy as NB
from sklearn.preprocessing import labelEncoder
from sklearn. Model_selection import train_test_split
```

2.2 DATA PREPROCESSING FUNCTIONS

Three function definitions are defined for the purpose of removing emojis, URL and whitespace. The remove_emoji function uses a python library called "re" [5] and compiles an emoji pattern [6], which is then applied over the testing and training set. The function for the same is shown below:

```
Def remove_EMOJI(text):
    Emoji_pat = re.compile("["
                           "u\U0001F600 -\U0001F64F"
                       "u\U0001F300 - \U0001F5FF"
                   "u\U0001F680 - \U0001F6FF"
                   "u\U0001F1E0 - \U0001F1FF"
                   "]" + flags = re.UNICODE)
    return Emoji_pat.sub(text)
```

In the above code snippet, the range of unicodes from "u\U0001F600 -\U0001F64F" is used as a pattern to match and remove the emotions from the dataset. The range "u\U0001F300 - \U0001F5FF" is used as pattern to match and remove the symbols and pictographs. The range "u\U0001F680 - \U0001F6FF" is used as the pattern to match and remove the transport and other symbols. The range "u\U0001F1E0 - \U0001F1FF" is used as a pattern to match and remove the IOS specific flags [7], which are crept in as a part gadget time stamps. The function for removing tweet specific URL is shown below.

```
def remove_URL(Text):
    URL_Pat = re.compile('http[s]?://[a-zA-Z] | [0-9] | [$_@.&+] | [?:%(0-9AF)]+')
    return URL_Pat.sub(text)
```


ISSN: 2582-337X
Submitted:08.06.2021
Revised: 09.06.2021
Accepted: 10.06.2021
Published: 14.06.2021

Journal of Ubiquitous Computing and Communication Technologies (UCCT) (2021)
Vol.03/ No.02
Pages: 61-79
https://www.irojournals.com/jucct/
DOI: https://doi.org/10.36548/jucct.2021.2.001

In the above function, an URL pattern ‘http[s]?://[a-zA-Z] | [0-9] | [$_@.&+] | [?:%(0-9AF)]+’ is used to check any tweet texts that start either with the pattern ‘http’ or ‘https’, followed by one or more occurrences of the character series ‘a’ to ‘z’ and ‘A’ to ‘Z’, numeral in the range 0-9 and any combination of special characters in the set {$_@.&+}. The pattern is then compiled and applied over the dataset for performing the initial trimming. The function for whitespace removal is defined as follows:

```
def remove_whitespace(text):
    del_dict = { sp_chr: ‘ ‘ for sp_chr in str.punctuation}
    del_dict[‘ ‘] = ‘ ‘
    tab1 = str.maketrans(del_dict)
    text1 = text.translate(tab1)
    textARR = text1.split()
    text2 = ‘’.join([w for w in textARR if (not isdigit() and len() > 1)])
    return text2.lower()
```

The above function creates a dictionary, which conforms to the pattern { sp_chr: ‘ ‘ for sp_chr in str.punctuation}. A list called tab1 is created to transform the input text based on the pattern sp_chr. The converted text is then split in to list with the limitation of length being more than one. The final text is returned in lower case as a part of the data preprocess.

After the preprocessing, it is noted from the dataset that the training set includes 8375 – “Postive”, 7673 – “negative” and 10704 – “neutral” tags from the total 26752 tuples. In the same way, the test set consists of 1075 – “positive”, 983 – “negative” and 1376 – “neutral” tuples. The figure 3 shows the output for the same.

ISSN: 2582-337X
Submitted:08.06.2021
Revised: 09.06.2021
Accepted: 10.06.2021
Published: 14.06.2021

Journal of Ubiquitous Computing and Communication Technologies (UCCT) (2021)
Vol.03/ No.02
Pages: 61-79
https://www.irojournals.com/jucct/
DOI: https://doi.org/10.36548/jucct.2021.2.001

```
test_data['text'] = test_data['text'].apply(remove_emoji)
test_data['text'] = test_data['text'].apply(remove_url)
test_data['text'] = test_data['text'].apply(clean_text)

 ==========Train Data =========
 neutral     10704
 positive     8375
 negative     7673
 Name: sentiment, dtype: int64
 26752
 ============================
 ==========Test Data =========
 neutral     1376
 positive    1075
 negative     983
 Name: sentiment, dtype: int64
 3434
 ============================
```

Figure 3. Wealth of dataset – Train and Test set

2.3 FITTING THE PREPROCESSED DATASET FOR THE CATEGORICAL CLASSIFICATION PREDICTION

The dataset so far processed consists of the tuples [8] in the form {sentiment_TAG: tuple record number}. This format is converted by using the package “LabelEncoder” to fit the transform on the text values, where the text values are “positive”, “negative” and “neutral”. The conversion yields the matching of “neutral” to 1, “positive” to 2 and “negative” to 0. This map of sentiment tag relationship needs to considered as “neutral” to [0, 1, 0], “positive” to [0, 0, 1] and “negative” to [1, 0, 0]. To perform the above the following code is used:

```
train_labels1 = le.fit_tranform(Train_data)
train_labels2 = np.nsarray(TF.Keras.utils.to_catogorical(train_labels1))
```

The figure 4 shows the outcome after performing the above steps.

ISSN: 2582-337X
Submitted:08.06.2021
Revised: 09.06.2021
Accepted: 10.06.2021
Published: 14.06.2021

Journal of Ubiquitous Computing and Communication Technologies (UCCT) (2021)
Vol.03/ No.02
Pages: 61-79
https://www.irojournals.com/jucct/
DOI: https://doi.org/10.36548/jucct.2021.2.001

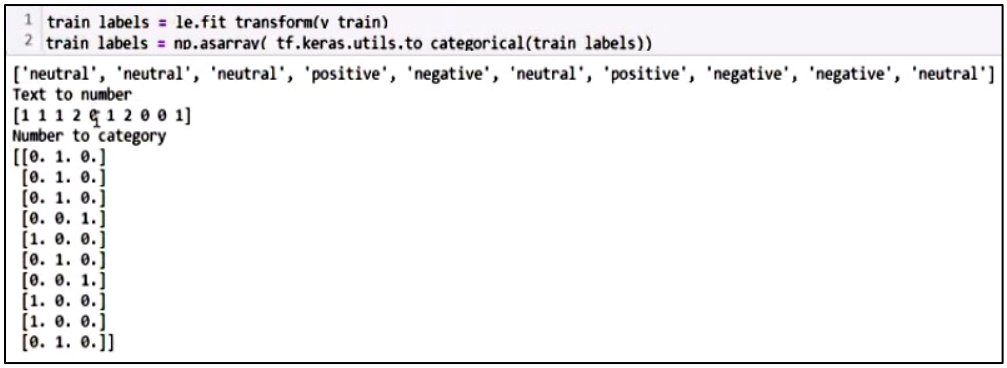

Figure 4. Categorical conversion of training data

3. NEURAL NETWORK MODEL

The first step in the neural network model is to convert the input text into vectors [9], such that the same is fed for the subsequent processing. This task is accomplished by the embedding layer, which is considered as first layer in the model. The embedding layer of the tensor flow model can perform the automated text pre-processing by choosing the best fitting methods. This is the casual method that leads to transfer learning. The layer utilized here is "/tf2_preview/gnews_swivel_20dm_with_oov" [10]. This layer is trainable by default. The code for embedding the layer is shown below:

```
Embed = "https://tfhub.dev/google/ tf2_preview/gnews_swivel_20dm_with_oov"
Embed_layer = tf_hub.kerasLayer(embedding, input_shape=[], dtype=tf.string,
trainable=True)
Embed_layer(Train_DS[:1])
```

The outcome of the Embedding layer is shown below in figure 5, where the input text is converted into a 20 dimensional vector, as we have chosen the layer to be 20dm one.

ISSN: 2582-337X
Submitted:08.06.2021
Revised: 09.06.2021
Accepted: 10.06.2021
Published: 14.06.2021

Journal of Ubiquitous Computing and Communication Technologies (UCCT) (2021)
Vol.03/ No.02
Pages: 61-79
https://www.irojournals.com/jucct/
DOI: https://doi.org/10.36548/jucct.2021.2.001

```
['just getting additional free']

<tf.Tensor: shape=(1, 20), dtype=float32, numpy=
array([[ 0.02252544, -1.102936  , -0.45312536,  1.1712923 , -0.9280152 ,
        -1.6799717 , -0.03097978, -0.04522766, -0.69689393, -0.3546787 ,
        -0.9819268 ,  1.3606595 , -0.7599448 ,  0.05979407, -0.71509284,
         0.48663723,  1.3430784 , -1.0147772 ,  0.13958323, -0.14541228]],
      dtype=float32)>
```

Figure 5. Input text to vector conversion

3.1 KERAS NEURAL NETWORK MODEL

Keras is a machine learning enhanced Application Programming Interface [API]. Keras APIs [11] are closely knitted in accordance with the Tensorflow 2.0 framework. This study has used the sequential model, which is considered as a collection of layers in plain stack. The model needs to be instantiated through the Keras framework. In this proposed model, the first layer is the "Embed_layer" [12]. The next layer is the dense hidden layer with fully interconnected neurons. This hidden layer is getting associated with the "ReLU" activation method, where "ReLU" [13] refers to Rectified Linear Unit. The next layer is also a fully interconnected dense layer with "Sigmoid" [14] as the activation method.

The model planned so far, is parsed to the Keras method called "compile", which actually creates the model. For training a neural network model, the loss function is needed. As the dataset under study is categorical (like, "Positive", "negative", "Neutral"), a function called "CategoricalCrossEntropy" [15] is used as the loss function. Every time the dataset is through with the loss function, a back-propagation is done to train the model until consistent and constant loss accuracy is witnessed. The back propagation and the subsequent entropy calculation use the optimizer called "adam" [16], which is an adoptive learning rate optimization algorithm. The codes for the above stages are shown below:

```
NN_Model1 =  TF.Keras.Sequential()
NN_Model.add(Embed_layer)
NN_Model.add(TF.Keras.layers.Dense(10, activation = "relu"))
```


ISSN: 2582-337X
Submitted:08.06.2021
Revised: 09.06.2021
Accepted: 10.06.2021
Published: 14.06.2021

Journal of Ubiquitous Computing and Communication Technologies (UCCT) (2021)
Vol.03/ No.02
Pages: 61-79
https://www.irojournals.com/jucct/
DOI: https://doi.org/10.36548/jucct.2021.2.001

NN_Model.add(TF.Keras.layers.dense(3, activation = “sigmoid”))

NN_Model.compile(loss = TF.Keras.losses.categoricalEntropy(Logits = True), optimizer = “adam”)

NN_Model.summary()

The outcome of the proposed neural network model summary is shown in figure X8. From the figure 6, it can be observed that, the Keras layer has weighted parameters of 389380 with 20 dimensional vectors. And it is ensured that, there are no non-trainable parameters. Thus, the tight connection of neural networks is also ensured.

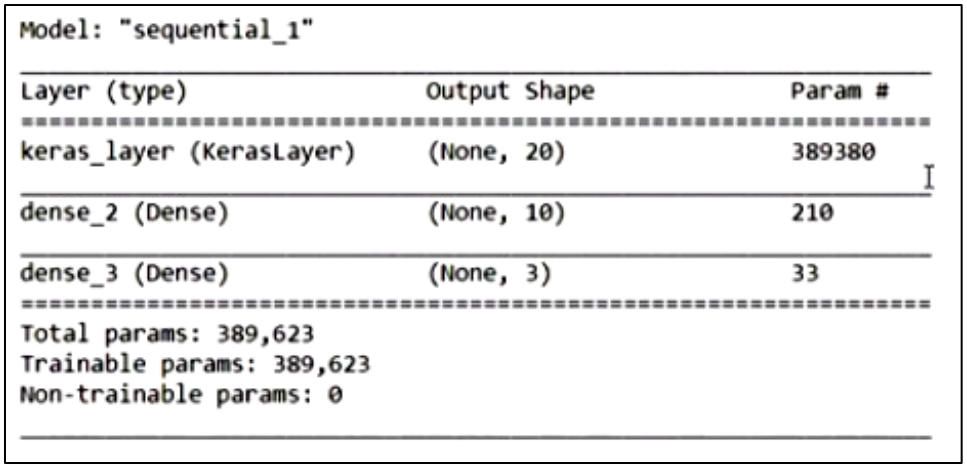

Figure 6. Tensorflow Neural Network Model Summary

The visual representation of the layers formed is depicted in the figure 7. From the representation it is observed that, the Keras input layer has no dimensionality association. This is depicted by ‘*’ in the outcome. Then, in the subsequent fabricated layers, the dimensionality is shown as 20, 10, and 3 for the Keras layer, dense RELU layer and dense sigmoid layer respectively. The visual representation is obtained by using the following code:

TF.Keras. Utils.Plot_model(NN_Model1, “Fig_X9.jpg”, show_shapes=True)

ISSN: 2582-337X
Submitted:08.06.2021
Revised: 09.06.2021
Accepted: 10.06.2021
Published: 14.06.2021

Journal of Ubiquitous Computing and Communication Technologies (UCCT) (2021)
Vol.03/ No.02
Pages: 61-79
https://www.irojournals.com/jucct/
DOI: https://doi.org/10.36548/jucct.2021.2.001

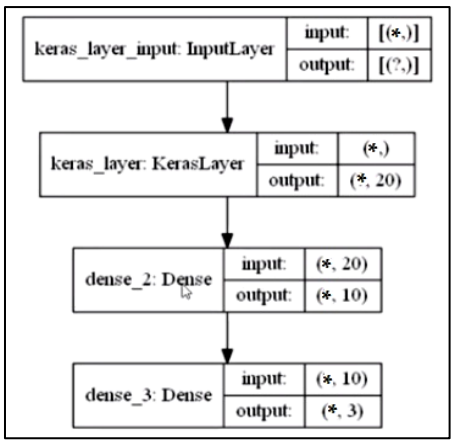

Figure 7. Visual layer wise representation of the neural network model

4. OPERATIONAL STAGE

4.1 TRAINING THE MODEL

For model training, a parameter called “Epochs” [17] is used and it is defined by giving the value, which is considered as the cycle / loop count value. In this study, after many trials, the value of 15 is assigned to the parameter called “Epochs”. Deviation in the count produces reduced prediction accuracy. The model.fit() function is used to start the fitting and subsequent processes. The code is shown below:

```
Outcome_Model_stage1 = model.fit(Train_DS.Shuffle(2000), batch(128), Epochs = 15, Verbose = 1)
```

The code has given the provision for shuffling with the value 2000. The batch size of shuffled units is 128. The iteration wise outcome is required for reference, accordingly the verbose property is set to 1. The outcome of the fitting is iterated 15 times and the history data is also collected. The figure 8 shows the details of the 15 iterations.

ISSN: 2582-337X
Submitted:08.06.2021
Revised: 09.06.2021
Accepted: 10.06.2021
Published: 14.06.2021

Journal of Ubiquitous Computing and Communication Technologies (UCCT) (2021)
Vol.03/ No.02
Pages: 61-79
https://www.irojournals.com/jucct/
DOI: https://doi.org/10.36548/jucct.2021.2.001

```
Train for 141 steps, validate for 69 steps
Epoch 1/20
141/141 [==============================] - 1s 8ms/step - loss: 1.0498 - CategoricalAccuracy: 0.4546 - val_loss: 0.9980 - val_
CategoricalAccuracy: 0.5673
Epoch 2/20
141/141 [==============================] - 1s 5ms/step - loss: 0.9364 - CategoricalAccuracy: 0.6481 - val_loss: 0.9361 - val_
CategoricalAccuracy: 0.6172
Epoch 3/20
141/141 [==============================] - 1s 5ms/step - loss: 0.8790 - CategoricalAccuracy: 0.6941 - val_loss: 0.9120 - val_
CategoricalAccuracy: 0.6301
Epoch 4/20
141/141 [==============================] - 1s 5ms/step - loss: 0.8503 - CategoricalAccuracy: 0.7178 - val_loss: 0.9010 - val_
CategoricalAccuracy: 0.6370
Epoch 5/20
141/141 [==============================] - 1s 5ms/step - loss: 0.8327 - CategoricalAccuracy: 0.7302 - val_loss: 0.8947 - val_
CategoricalAccuracy: 0.6399
Epoch 6/20
141/141 [==============================] - 1s 5ms/step - loss: 0.8174 - CategoricalAccuracy: 0.7389 - val_loss: 0.8904 - val

Epoch 11/20
141/141 [==============================] - 1s 5ms/step - loss: 0.7755 - CategoricalAccuracy: 0.7779 - val_loss: 0.8850 - val_
CategoricalAccuracy: 0.6442
Epoch 12/20
141/141 [==============================] - 1s 5ms/step - loss: 0.7709 - CategoricalAccuracy: 0.7830 - val_loss: 0.8858 - val_
CategoricalAccuracy: 0.6437
Epoch 13/20
141/141 [==============================] - 1s 5ms/step - loss: 0.7672 - CategoricalAccuracy: 0.7873 - val_loss: 0.8858 - val_
CategoricalAccuracy: 0.6436
Epoch 14/20
141/141 [==============================] - 1s 5ms/step - loss: 0.7591 - CategoricalAccuracy: 0.7912 - val_loss: 0.8862 - val_
CategoricalAccuracy: 0.6437
Epoch 15/20
141/141 [==============================] - 1s 5ms/step - loss: 0.7536 - CategoricalAccuracy: 0.7955 - val_loss: 0.8863 - val_
CategoricalAccuracy: 0.6436
```

Figure 8. The Epoch cycles from 1 to 15

From the figure it is notable that during Epoch = 1, the categorical accuracy is observed as 0.4546 and the validation accuracy as 0.5673. The accuracy of the later one is much higher than the former one. Also at the end of 15th Epoch, the categorical accuracy has been higher than the validation accuracy. The loss function starts with the value of 1.0498 and at the end of 15th Epoch the loss is decreased by 0.7536. But it is also noted that, in the validation lose, the values are not following any pattern (Neither ascending nor descending). Pictorially, this data is plotted between the epoch count and the loss accuracy over the training data. The figure 9 shows the plot.

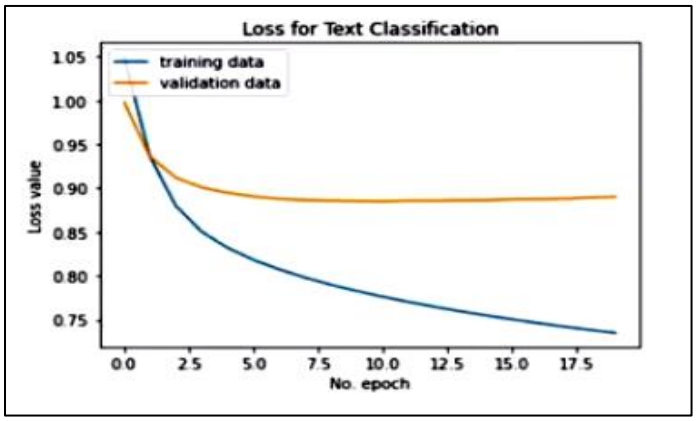

Figure 9. Loss for text classification between Epoch Vs. Loss accuracy (Over training data)

ISSN: 2582-337X
Submitted:08.06.2021
Revised: 09.06.2021
Accepted: 10.06.2021
Published: 14.06.2021

Journal of Ubiquitous Computing and Communication Technologies (UCCT) (2021)
Vol.03/ No.02
Pages: 61-79
https://www.irojournals.com/jucct/
DOI: https://doi.org/10.36548/jucct.2021.2.001

From the figure it is noted that, the loss for the validation data does not decrease when compared with the training data. There is a huge gap in the linearity between two components. These types of models are called as the over-fitting models. Similarly, the plot for categorical accuracy between the categorical loss and epoch is plotted and shown in figure 10.

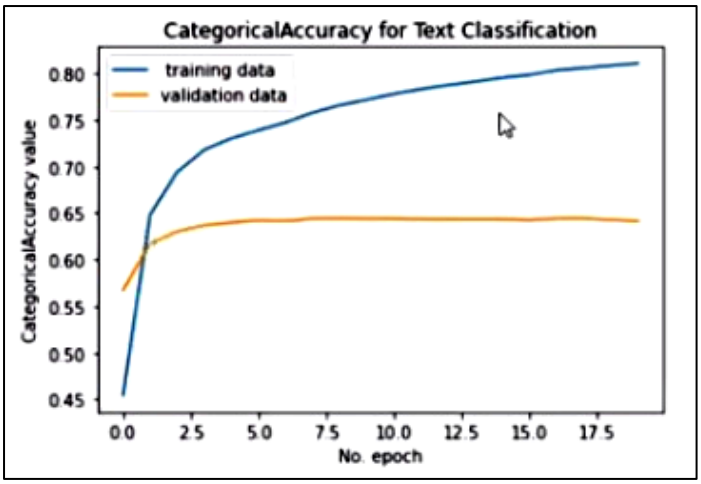

Figure 10. Loss accuracy for categorical loss between Epoch and loss accuracy

Figure 12 also confirms that, the model definitely over fits the data by degrading the prediction efficiency.

4.2 ACCURACY OPTIMIZATION TECHNIQUE

The techniques deployed for optimizing the accuracy are drop-out [18] and Kernel regularizes [19]. Drop-out is a deep neural network version, which randomly sets few of the misleading records to zero. Thus it encourages each node to carry out the independency into manifold levels. This model makes the “1-rate” optimality to its maximum available in the dataset.

On the other hand, the Kernel regularizers assign the penalty on the layers by ensuring the smooth functioning of layer activities. These penalty parameters are also optimized at the later stage by using the same kernel regularizers for housekeeping activity. As we have L1 and L2

ISSN: 2582-337X
Submitted:08.06.2021
Revised: 09.06.2021
Accepted: 10.06.2021
Published: 14.06.2021

Journal of Ubiquitous Computing and Communication Technologies (UCCT) (2021)
Vol.03/ No.02
Pages: 61-79
https://www.irojournals.com/jucct/
DOI: https://doi.org/10.36548/jucct.2021.2.001

level of kernel regularizers along with their pros and cons, this model utilizes both of them. The L1 regularizer and the L2 regularizer are applied here. Basically L1 combined with L2 is termed as LASO->RIDGE->Elastic Net progression [20]. The code for the above modification is shown below:

```
NN_Model1 = TF.Keras. Sequential()
Model.add(NN_Model1)
NN_model1.add(TF.Keras.layers.Dropout(0.5))
NN_model1.add(TF.Keras.layers.Dense(10, activation=”relu”, Kernel_regularizer=
regularizer.L1))
NN_model1.add(TF.Keras.layers.Dropout(0.5))
NN_model1.add(TF.Keras.layers.Dense(10, activation=”relu”, Kernel_regularizer=
regularizer.L2))
```

This improvement is depicted in figure 11.Once again; the correlation accuracy outcome has been depicted in figure 12. This figure shows the improvement in accuracy.

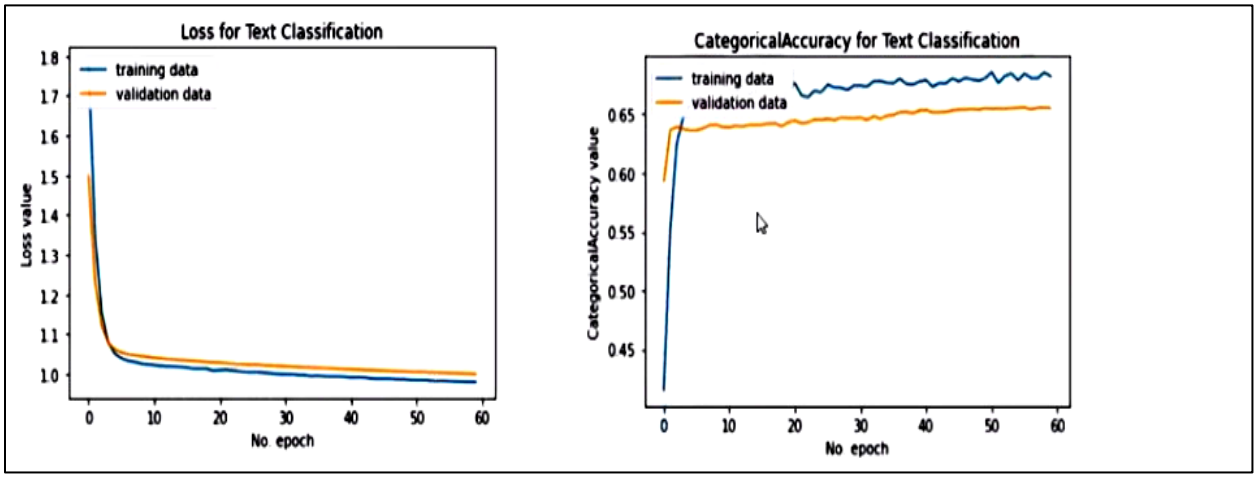

Figure 11. The correlation Plot after the Inclusion of Regularizers

ISSN: 2582-337X
Submitted:08.06.2021
Revised: 09.06.2021
Accepted: 10.06.2021
Published: 14.06.2021

Journal of Ubiquitous Computing and Communication Technologies (UCCT) (2021)
Vol.03/ No.02
Pages: 61-79
https://www.irojournals.com/jucct/
DOI: https://doi.org/10.36548/jucct.2021.2.001

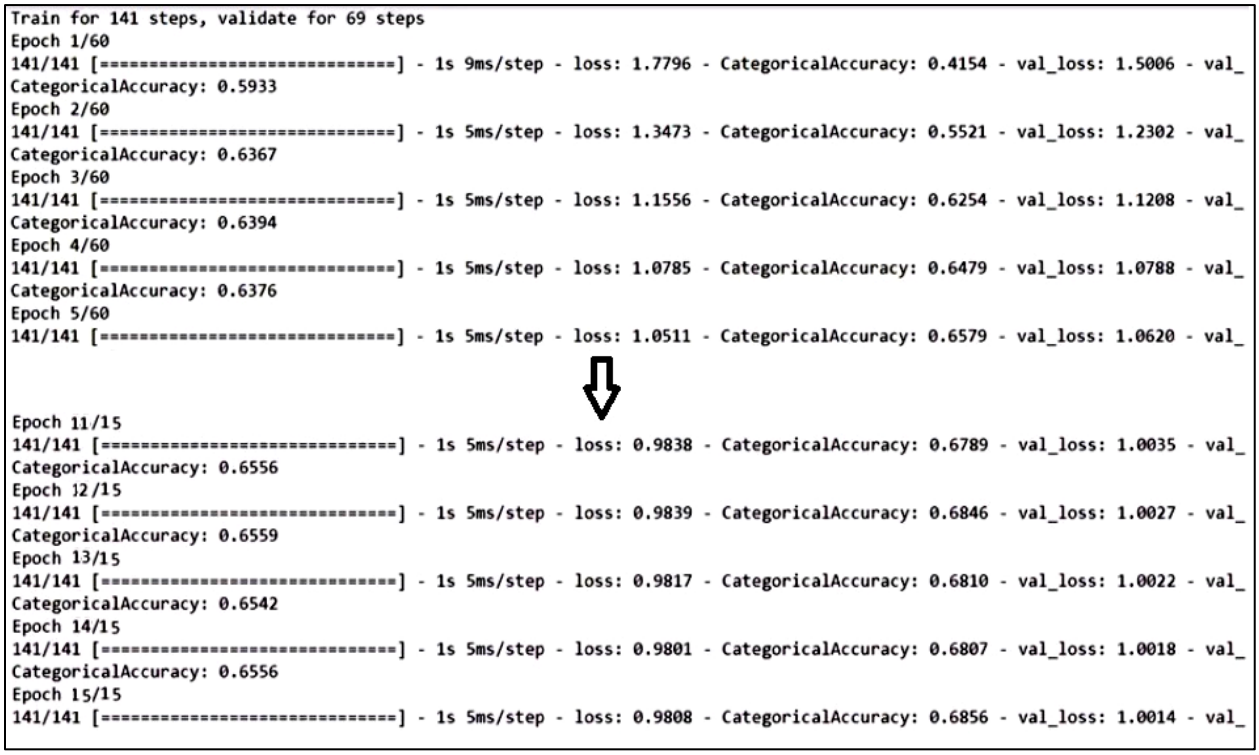

Figure 12. Accuracy Improvement after the Inclusion of Regularizers

CONCLUSION

In this study, a neural network model is deployed over the tensor flow framework. The model is developed in layers with the activation methods namely ReLU and Sigmoid. The categorical loss accuracy is measured and it is found that, the model is over-fitting and eventually it will result in degrading the prediction accuracy. Then, the model is converted in to a deep neural network using the drop-out optimizers and Kernel regularizers. The model is once again trained for 15 Epochs and the values are fetched accordingly. The resultant value shows an optimized improvement in the prediction accuracy based on the available dataset.

The proposed study has captured the accuracy outcomes of training loss and test loss based on the kaggle dataset. Initially, it is identified that, the loss accuracy is over-fitting the prediction. The inclusion of “drop-out” and “Kernel regularizer” layers to the existing neural network has improvised the underlying model to a deep neural network. The Epoch count of 15, which is fixed based on trial-and-error. This has enhanced the prediction accuracy to the extent of

ISSN: 2582-337X
Submitted:08.06.2021
Revised: 09.06.2021
Accepted: 10.06.2021
Published: 14.06.2021

Journal of Ubiquitous Computing and Communication Technologies (UCCT) (2021)
Vol.03/ No.02
Pages: 61-79
https://www.irojournals.com/jucct/
DOI: https://doi.org/10.36548/jucct.2021.2.001

additional 25% when compared to the existing neural network layers. The future research scope would be to test the model with dynamic contents.

REFERENCES

[1] Di Gangi, Mattia A., et al. "MuST-C: a multilingual speech translation corpus." 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics, 2019.

[2] Lnenicka, Martin, and Jitka Komarkova. "Big and open linked data analytics ecosystem: Theoretical background and essential elements." Government Information Quarterly 36.1 (2019): 129-144.

[3] Sungheetha, Akey, and Rajesh Sharma. "A Comparative Machine Learning Study on IT Sector Edge Nearer to Working From Home (WFH) Contract Category for Improving Productivity." Journal of Artificial Intelligence 2, no. 04 (2020): 217-225.

[4] Smilkov, Daniel, Nikhil Thorat, Yannick Assogba, Ann Yuan, Nick Kreeger, Ping Yu, Kangyi Zhang et al. "Tensorflow. js: Machine learning for the web and beyond." arXiv preprint arXiv:1901.05350 (2019).

[5] Manoharan, Samuel. "Embedded Imaging System Based Behavior Analysis of Dairy Cow." Journal of Electronics 2, no. 02 (2020): 148-154.

[6] Chakrabarty, Navoneel, and Sanket Biswas. "Navo Minority Over-sampling Technique (NMOTe): A Consistent Performance Booster on Imbalanced Datasets." Journal of Electronics 2, no. 02 (2020): 96-136.

[7] S. Thivaharan., G. Srivatsun. and S. Sarathambekai., "A Survey on Python Libraries Used for Social Media Content Scraping," 2020 International Conference on Smart Electronics and

ISSN: 2582-337X
Submitted:08.06.2021
Revised: 09.06.2021
Accepted: 10.06.2021
Published: 14.06.2021

Communication (ICOSEC), Trichy, India, 2020, pp. 361-366, doi: 10.1109/ICOSEC49089.2020.9215357.

[8] Sharma, Uday Bhaskar, and Anupam Singh. "Commuting probability and simultaneous conjugacy classes of commuting tuples in a group." arXiv preprint arXiv:2002.01253 (2020).

[9] Rodman, Emma. "A Timely Intervention: Tracking the Changing Meanings of Political Concepts with Word Vectors." Political Analysis 28.1 (2020): 87-111.

[10] Lambert, John, et al. "MSeg: A Composite Dataset for Multi-domain Semantic Segmentation." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020.

[11] Awan, Ammar Ahmad, et al. "HyPar-Flow: Exploiting MPI and Keras for Scalable Hybrid-Parallel DNN Training using TensorFlow." arXiv preprint arXiv:1911.05146 (2019).

[12] Adam, Edriss Eisa Babikir. "Deep Learning based NLP Techniques In Text to Speech Synthesis for Communication Recognition." Journal of Soft Computing Paradigm (JSCP) 2, no. 04 (2020): 209-215..

[13] Zou, Difan, et al. "Gradient descent optimizes over-parameterized deep ReLU networks." Machine Learning 109.3 (2020): 467-492.

[14] Goel, Priyanka, and S. Sivaprasad Kumar. "Certain class of starlike functions associated with modified sigmoid function." Bulletin of the Malaysian Mathematical Sciences Society 43, no. 1 (2020): 957-991.

[15] Chakraborty, Rupak, Rama Sushil, and M. L. Garg. "An improved PSO-based multilevel image segmentation technique using minimum cross-entropy thresholding." Arabian Journal for Science and Engineering 44.4 (2019): 3005-3020.

ISSN: 2582-337X
Submitted:08.06.2021
Revised: 09.06.2021
Accepted: 10.06.2021
Published: 14.06.2021

Journal of Ubiquitous Computing and Communication Technologies (UCCT) (2021)
Vol.03/ No.02
Pages: 61-79
https://www.irojournals.com/jucct/
DOI: https://doi.org/10.36548/jucct.2021.2.001

[16] Hamdan, Yasir Babiker. "Faultless Decision Making for False Information in Online: A Systematic Approach." Journal of Soft Computing Paradigm (JSCP) 2, no. 04 (2020): 226-235.

[17] Siddique, Fathma, Shadman Sakib, and Md Abu Bakr Siddique. "Recognition of handwritten digit using convolutional neural network in python with tensorflow and comparison of performance for various hidden layers." 2019 5th International Conference on Advances in Electrical Engineering (ICAEE). IEEE, 2019.

[18] Moreno-Marcos, Pedro Manuel, et al. "Temporal analysis for dropout prediction using self-regulated learning strategies in self-paced MOOCs." Computers & Education 145 (2020): 103728.

[19] Li, Zhu, et al. "Kernel dependence regularizers and gaussian processes with applications to algorithmic fairness." arXiv preprint arXiv:1911.04322 (2019).

[20] Pooja.C, Thivaharan.s, "Workload based Cluster Auto Scaler using Kuberbet Monitors", International Journal Compliance Engineering Journal (IJCENG), 2021, Vol.12, Issue.6, pp. 40-47, ISSN:0898-3577, DOI:16.10089.CEJ.2021.V12I6.285311.36007.

AUTHORS BIOGRAPHY

[1]Mr. S. Thivaharan did his B.Tech (IT) from Thiagarajar College of Engineering. He completed his M.Tech in Networking from Amrita School of Engineering. He is presently pursuing PhD in Anna University, Chennai. He worked as a system Engineer in TATA Consultancy Services Ltd, Chennai for 3 years. Then he switched to teaching profession. He is presently working as Assistant Professor (Selection Grade) in PSG Institute of Technology and Applied Research, Coimbatore. He is having a total of 12 years of teaching experience.

ISSN: 2582-337X
Submitted:08.06.2021
Revised: 09.06.2021
Accepted: 10.06.2021
Published: 14.06.2021

Journal of Ubiquitous Computing and Communication Technologies (UCCT) (2021)
Vol.03/ No.02
Pages: 61-79
https://www.irojournals.com/jucct/
DOI: https://doi.org/10.36548/jucct.2021.2.001

[2]Dr. G. Srivatsun did his BE (ECE) from Bharathiyar University. He Completed his ME (Wireless technologies) from Anna University and PhD in the area of RF antennas from Anna University, Chennai. After a year of service from affiliated institution, he joined PSG college of Technology. He has won the AICTE - Career award for young teachers during the year 2013 for his research in antennas, through which he has traveled to USA. he has published various national and international journals. He has also served as faculty advisor for student affairs, coordinated the dean functioning and research activities. He is the nodal officer for All India Survey for Higher Education (MHRD).

ISSN: 2582-337X
Submitted:08.06.2021
Revised: 09.06.2021
Accepted: 10.06.2021
Published: 14.06.2021