TOXIC COMMENTS CALSSIFICATION USING NLP

K. Shanthan Reddy, L. Akhil Reddy

Department of Computer Science and Engineering, SR University, Warangal, Telangana, India.
Department of Computer Science and Engineering, SR University, Warangal, Telangana, India.

EMAIL: kundurushanthan777@gmail.com
lakhilreddy20@gmail.com

ABSTRACT:

Over the past ten years, comment analysis have seen significant growth. Users mental plays a crucial in the activate participation in day to day social media activities. Negative comments plays a negative role in user's mental health and performance. Nowadays users leave numerous comments on different social networks, news portals, and forums. Some of the comments are toxic or abusive. Due to numbers of comments, it is a lot of work to manually moderate all of the comments. So we have taken an natural language processing as core subject and took Lstm model to train the comments and universal sentence encoder to embed the sentences. The dataset we used have a large amount of toxic and non toxic comments , but by using this model we were able to predict if that comment is toxic or not and what is the toxicity of that comment and by using the lstm model for this project made us get a good accuracy. The dataset we took all was compatible with the model. So, we got any accuracy rate of 86

Keywords: LSTM, universal sentence encoder

1. INTRODUCTION

In the early days of the Internet, people communicated only by e-mail that encountered spam. At the time, classifying emails as good or bad, whether or not spam was a challenge. Internet communications and data flows have changed significantly over time, especially since the development of social networks. With the advent of social media, classifying content as “good” or “bad” has become more important than ever to prevent social harm and prevent people from engaging in antisocial behavior.

Internet is an open discussing space for anyone to express their opinions. We can express anything whatever we need and there is no classification in the comments. However the use of internet has been more in the recent days and there are many number of platforms where we can express our feeling without any restrictions. Also the use of internet has been very useful in many ways which helps us in learning.

Motivated by this problem, we want to build a platform where we can know the amount of toxicity present in a sentence. Where toxicity is defined as anything that is rude, disrespectful or any otherwise, the sentence that is likely to make someone leave a discussion.

Because toxic comments limit people's ability to express themselves and have different opinions, as a result of the negative, there are no positive conversations on social networks. As a result, detecting and restricting antisocial behavior in online discussion forums is a pressing requirement. These toxic comments might be offensive, menacing, or disgusting. Our goal is to identify these harmful remarks.

2.PROBLEM DEFINATION:

Identifying the toxic comments can help in many ways and help remove them from that platform

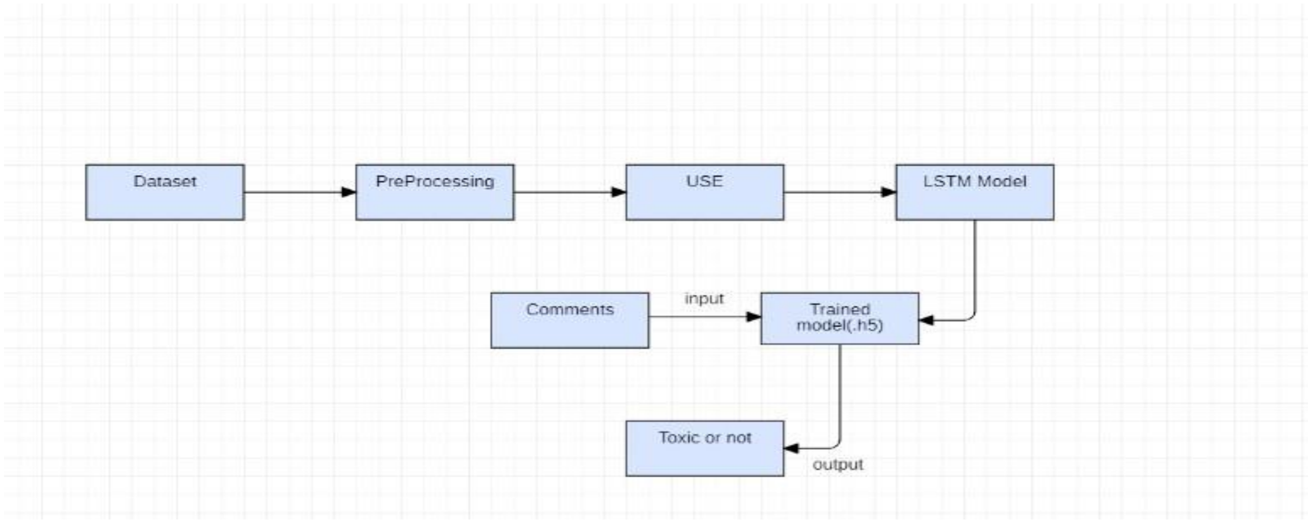


Figure 1. processing of toxic comments

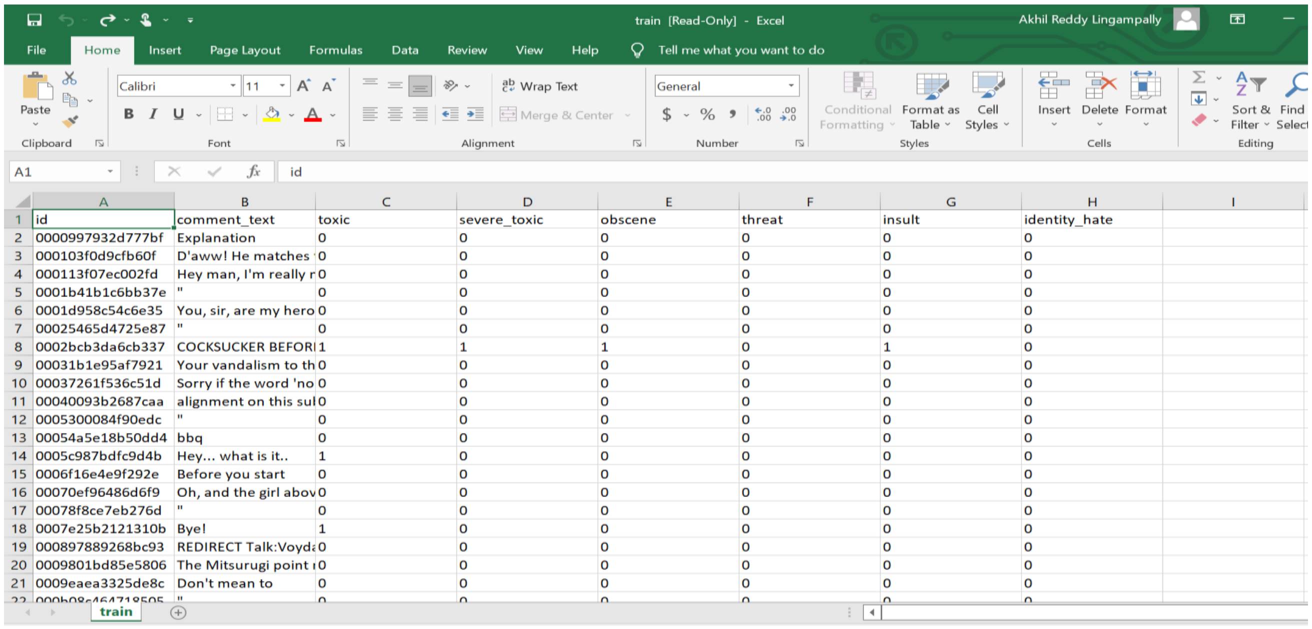
3.DATASET:

We have collected our dataset from Kaggle. The data we collected has 8 columns and 150000+ rows . but for our model we need only 2 columns so out of 8 columns we consider only 2 columns. Also we do not need all the so we train the model only using the 2 columns. The model to be trained with over 150000+ rows has more execution time so we lower it to 10000 and we train the model. It is developed by jigsaw. The comments in the data are 90% non toxic and 10% toxic.(we got this while training the dataset).

Columns in dataset:

- id
- comment_text
- toxic
- severe_toxic
- obscene
- threat
- insult
- identify_hate

From the above columns we will use only comment_text, toxic.



	A	B	C	D	E	F	G	H	I
1	id	comment_text	toxic	severe_toxic	obscene	threat	insult	identity_hate	
2	0000997932d777bf	Explanation	0	0	0	0	0	0	
3	000103f0d9cfb60f	D'aww! He matches	0	0	0	0	0	0	
4	000113f07ec002fd	Hey man, I'm really r	0	0	0	0	0	0	
5	0001b41b1c6bb37e	"	0	0	0	0	0	0	
6	0001d958c54c6e35	You, sir, are my hero	0	0	0	0	0	0	
7	00025465d4725e87	"	0	0	0	0	0	0	
8	0002bcb3da6cb337	COCKSUCKER BEFORI	1	1	1	0	1	0	
9	00031b1e95af7921	Your vandalism to th	0	0	0	0	0	0	
10	00037261f536c51d	Sorry if the word 'no	0	0	0	0	0	0	
11	00040093b2687caa	alignment on this sul	0	0	0	0	0	0	
12	0005300084f90edc	"	0	0	0	0	0	0	
13	00054a5e18b50dd4	bbq	0	0	0	0	0	0	
14	0005c987bdfc9d4b	Hey... what is it..	1	0	0	0	0	0	
15	0006f16e4e9f292e	Before you start	0	0	0	0	0	0	
16	00070ef96486d6f9	Oh, and the girl abov	0	0	0	0	0	0	
17	00078f8ce7eb276d	"	0	0	0	0	0	0	
18	0007e25b2121310b	Bye!	1	0	0	0	0	0	
19	000897889268bc93	REDIRECT Talk:Voyd	0	0	0	0	0	0	
20	0009801bd85e5806	The Mitsurugi point	0	0	0	0	0	0	
21	0009eaea3325de8c	Don't mean to	0	0	0	0	0	0	

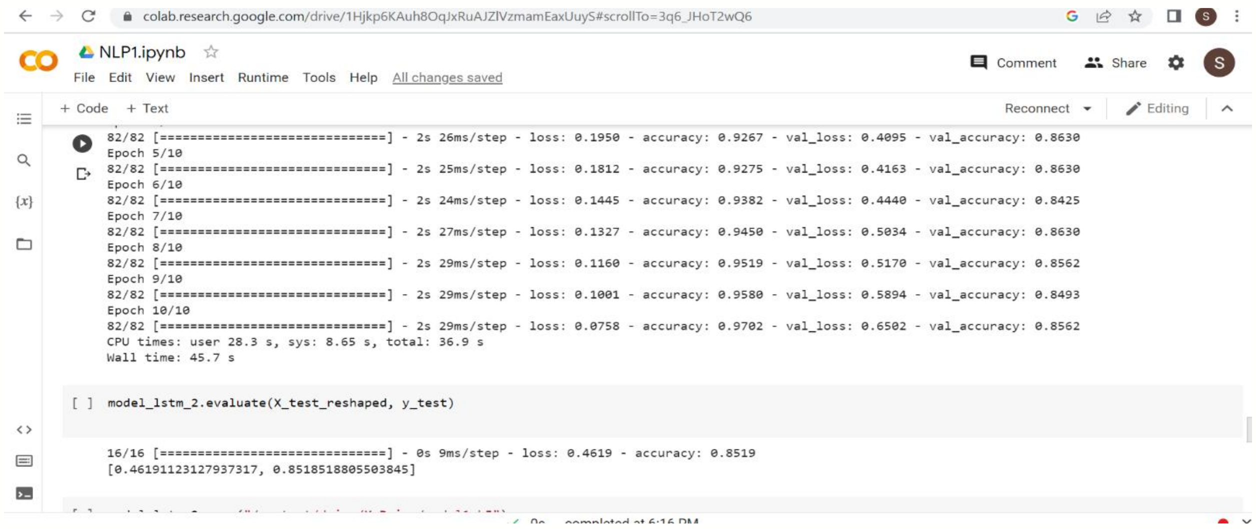
Figure 2 : data

4.DATA PREPROCESSING:

First we are going to install the tensor flow 2.6.0 version rather than new version, because we are getting some errors in our project for the latest version.th

```
%%capture
!pip install tensorflow_text==2.6.0
```

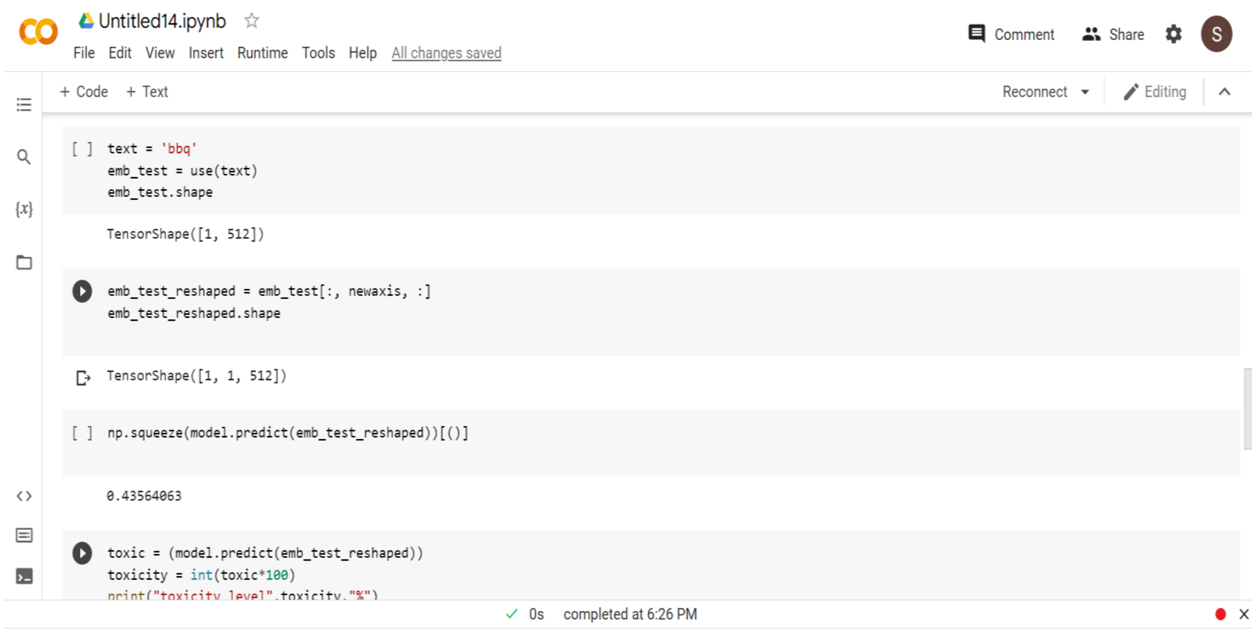
Then we are going to import all the necessary libraries required for the project.
We will be doing this project in google collab. We are going to import the dataset from the google drive of kundurushanthan777@gmail.com.Since the dataset is large it is going to take more amount of time train and test the dataset. Since the dataset contains 150000+ lines. So, in the next step we are going to reduce the size of the rows to 10000 only to easily test and train the data.


colab.research.google.com/drive/1Hjkp6KAuh8OqJxRuAJZlVzmamEaxUuyS#scrollTo=3q6_JHoT2wQ6
NLP1.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
Reconnect Editing

```
82/82 [==============================] - 2s 26ms/step - loss: 0.1950 - accuracy: 0.9267 - val_loss: 0.4095 - val_accuracy: 0.8630
Epoch 5/10
82/82 [==============================] - 2s 25ms/step - loss: 0.1812 - accuracy: 0.9275 - val_loss: 0.4163 - val_accuracy: 0.8630
Epoch 6/10
82/82 [==============================] - 2s 24ms/step - loss: 0.1445 - accuracy: 0.9382 - val_loss: 0.4440 - val_accuracy: 0.8425
Epoch 7/10
82/82 [==============================] - 2s 27ms/step - loss: 0.1327 - accuracy: 0.9450 - val_loss: 0.5034 - val_accuracy: 0.8630
Epoch 8/10
82/82 [==============================] - 2s 29ms/step - loss: 0.1160 - accuracy: 0.9519 - val_loss: 0.5170 - val_accuracy: 0.8562
Epoch 9/10
82/82 [==============================] - 2s 29ms/step - loss: 0.1001 - accuracy: 0.9580 - val_loss: 0.5894 - val_accuracy: 0.8493
Epoch 10/10
82/82 [==============================] - 2s 29ms/step - loss: 0.0758 - accuracy: 0.9702 - val_loss: 0.6502 - val_accuracy: 0.8562
CPU times: user 28.3 s, sys: 8.65 s, total: 36.9 s
Wall time: 45.7 s
```

```
[ ] model_lstm_2.evaluate(X_test_reshaped, y_test)
```

```
16/16 [==============================] - 0s 9ms/step - loss: 0.4619 - accuracy: 0.8519
[0.46191123127937317, 0.8518518805503845]
```


Untitled14.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
Reconnect Editing
✓ 0s completed at 6:26 PM

```
[ ] text = 'bbq'
    emb_test = use(text)
    emb_test.shape
```

```
TensorShape([1, 512])
```

```
emb_test_reshaped = emb_test[:, newaxis, :]
emb_test_reshaped.shape
```

```
TensorShape([1, 1, 512])
```

```
[ ] np.squeeze(model.predict(emb_test_reshaped))[()]
```

```
0.43564063
```

```
toxic = (model.predict(emb_test_reshaped))
toxicity = int(toxic*100)
print("toxicity level",toxicity,"%")
```

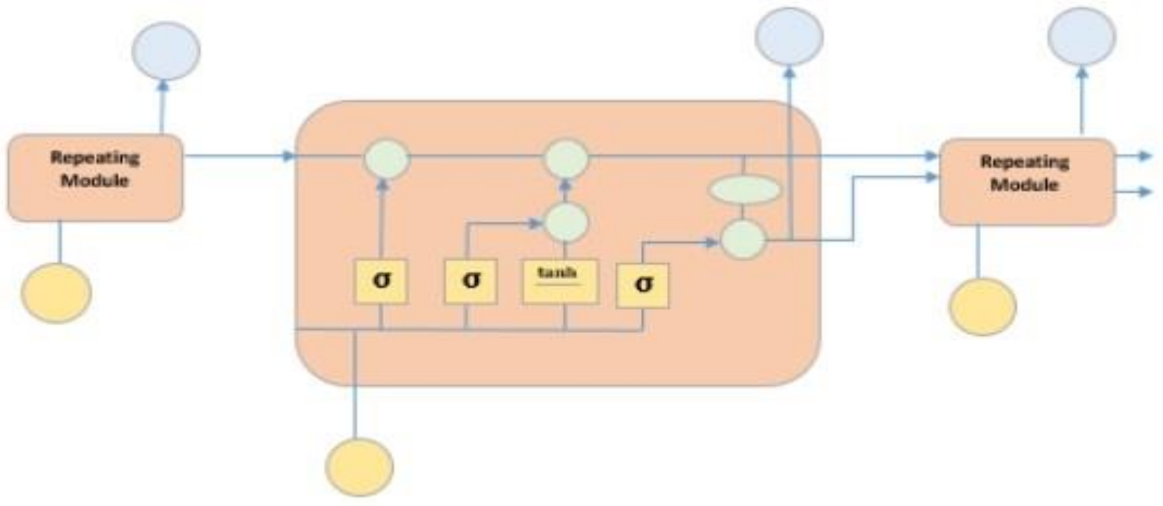
5. METHODOLOGY:

LSTM (Long Short-Term Memory):

Long short-term memory (LSTM) is an artificial neural network used in the fields of artificial intelligence and deep learning. Unlike standard feedforward neural networks, LSTM has feedback connections. Such a recurrent neural network (RNN) can process not only single data points (such as images), but also entire sequences of data (such as speech or video). For example, LSTM is applicable to tasks such as unsegmented, connected handwriting recognition, speech recognition, machine translation, robot control, video games, and healthcare. LSTM has become the most cited neural network of the 20th century.

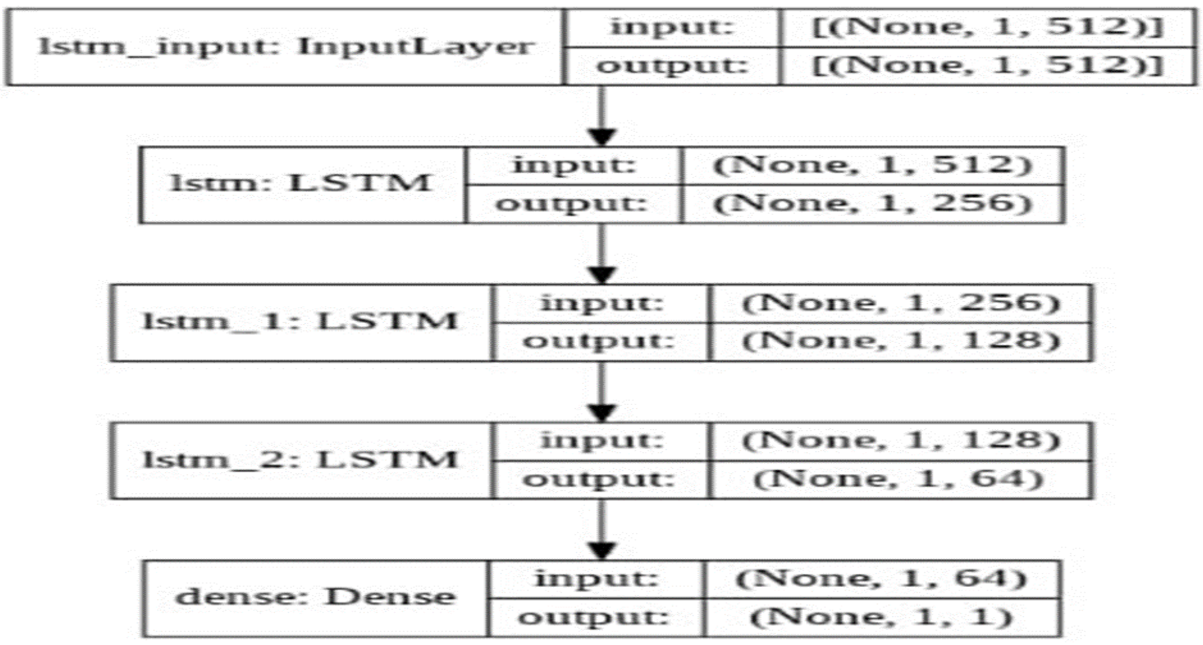
The name of LSTM refers to the analogy that a standard RNN has both "long-term memory" and "short-term memory". The connection weights and biases in the network change once per episode of training, analogous to how physiological changes in synaptic strengths store long-term memories; the activation patterns in the network change once per time-step, analogous to how the moment-to-moment change in electric firing patterns in the brain store short-term memories. The LSTM architecture aims to provide a short-term memory for RNN that can last thousands of timesteps, thus "long short-term memory".

It is special kind of recurrent neural network that is capable of learning long term dependencies in data. This is achieved because the recurring module of the model has a combination of four layers interacting with each other.



The picture above depicts four neural network layers in yellow boxes, point wise operators in green circles, input in yellow circles and cell state in blue circles. An LSTM module has a cell state and three gates which provides them with the power to selectively learn, unlearn or retain information from each of the units. The cell state in LSTM helps the information to flow through the units without being altered by allowing only a few linear interactions.

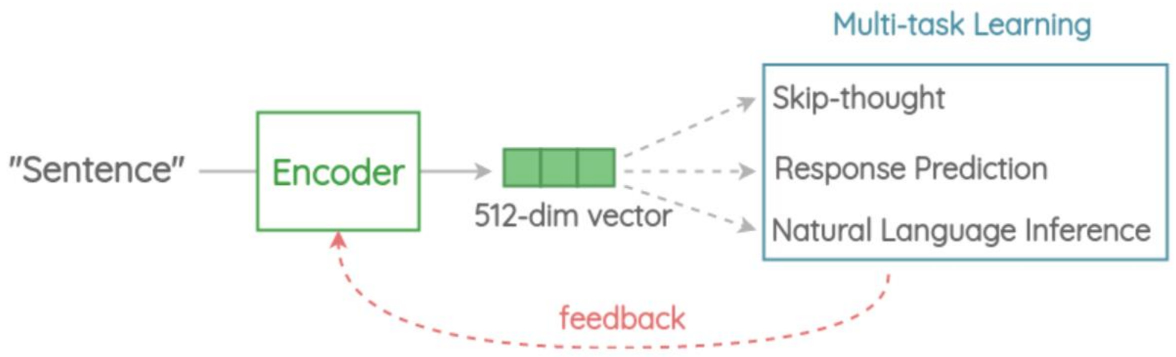
Model Architecture:



Universal Sentence Encoder:

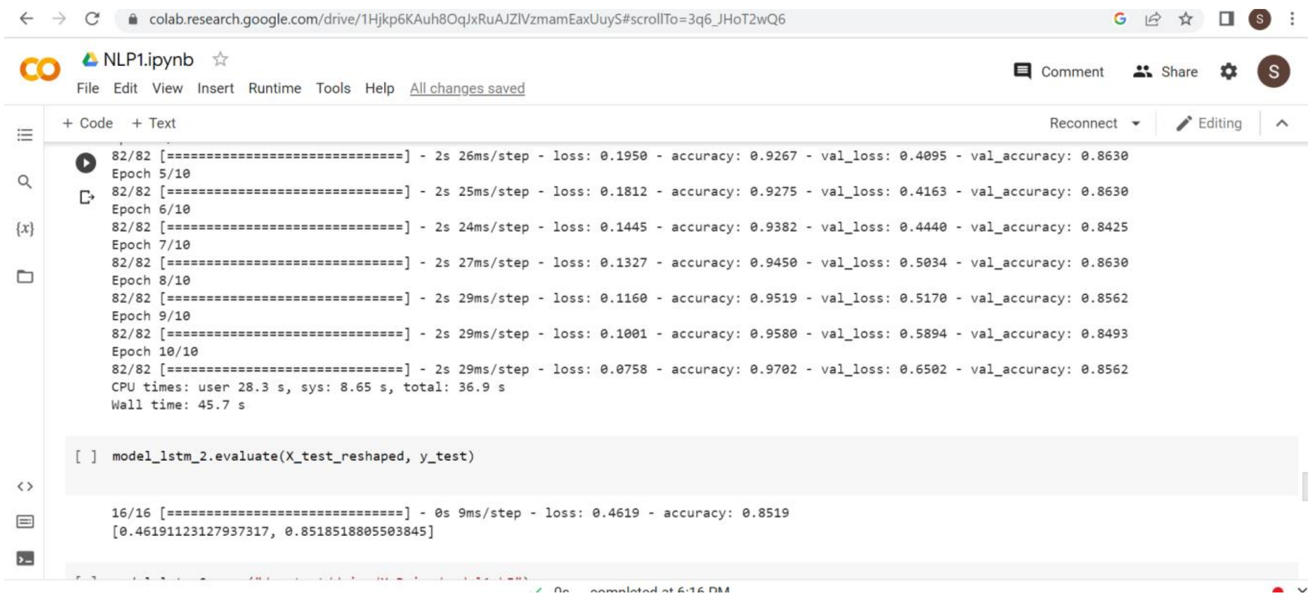
The universal sentence encoder makes looking up embeddings at the sentence level as simple as it has previously been to look up embeddings at the word level. Then, using less supervised training data, the sentence embeddings can be easily employed to compute sentencelevel meaning similarity and improve performance on subsequent classification tasks. The universal sentence encoder model converts textual information into numerically represented, high-dimensional vectors called embeddings. It aims to transfer learning especially to other NLP tasks like text categorization, semantic similarity, and clustering. The freely accessible universal sentence encoder is listed in Tensor flow-hub. To learn for a wide range of jobs, it is trained on a number of data sources.

On a high level, the idea is to design an encoder that summarizes any given sentence to a 512-dimensional sentence embedding. We use this same embedding to solve multiple tasks and based on the mistakes it makes on those, we update the sentence embedding. Since the same embedding has to work on multiple generic tasks, it will capture only the most informative features and discard noise. The intuition is that this will result in an generic embedding that transfers universally to wide variety of NLP tasks such as relatedness, clustering, paraphrase detection and text classification.



6.RESULTS:

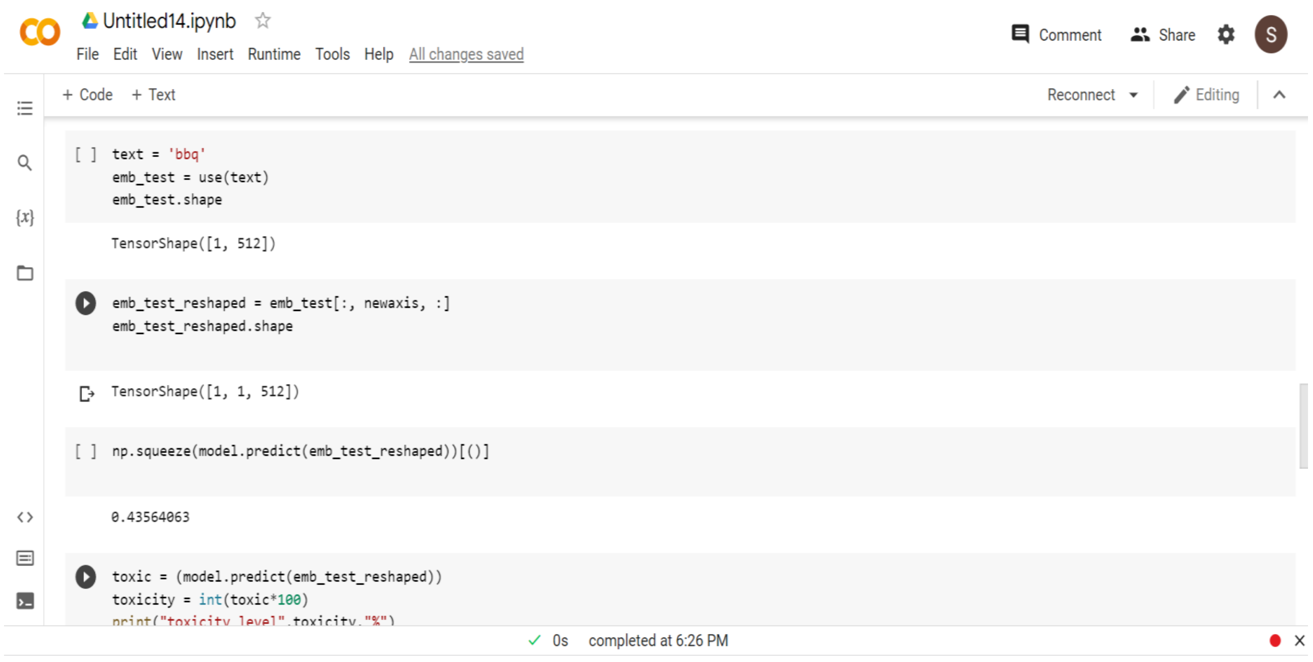
From the above project we have done we got an accuracy rate of 86% and we are able to the toxicity percentage of the given text. We are also showing the if the comment is good or bad. So by using lstm model only we are getting this of accuracy rate.


colab.research.google.com/drive/1Hjkp6KAuh8OqJxRuAJZlVzmamEaxUuyS#scrollTo=3q6_JHoT2wQ6
NLP1.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
Reconnect
Editing

```
82/82 [==============================] - 2s 26ms/step - loss: 0.1950 - accuracy: 0.9267 - val_loss: 0.4095 - val_accuracy: 0.8630
Epoch 5/10
82/82 [==============================] - 2s 25ms/step - loss: 0.1812 - accuracy: 0.9275 - val_loss: 0.4163 - val_accuracy: 0.8630
Epoch 6/10
82/82 [==============================] - 2s 24ms/step - loss: 0.1445 - accuracy: 0.9382 - val_loss: 0.4440 - val_accuracy: 0.8425
Epoch 7/10
82/82 [==============================] - 2s 27ms/step - loss: 0.1327 - accuracy: 0.9450 - val_loss: 0.5034 - val_accuracy: 0.8630
Epoch 8/10
82/82 [==============================] - 2s 29ms/step - loss: 0.1160 - accuracy: 0.9519 - val_loss: 0.5170 - val_accuracy: 0.8562
Epoch 9/10
82/82 [==============================] - 2s 29ms/step - loss: 0.1001 - accuracy: 0.9580 - val_loss: 0.5894 - val_accuracy: 0.8493
Epoch 10/10
82/82 [==============================] - 2s 29ms/step - loss: 0.0758 - accuracy: 0.9702 - val_loss: 0.6502 - val_accuracy: 0.8562
CPU times: user 28.3 s, sys: 8.65 s, total: 36.9 s
Wall time: 45.7 s
```

```
[ ] model_lstm_2.evaluate(X_test_reshaped, y_test)
```

```
16/16 [==============================] - 0s 9ms/step - loss: 0.4619 - accuracy: 0.8519
[0.46191123127937317, 0.8518518805503845]
```



```
text = 'bbq'
emb_test = use(text)
emb_test.shape
```

```
TensorShape([1, 512])
```

```
emb_test_reshaped = emb_test[:, newaxis, :]
emb_test_reshaped.shape
```

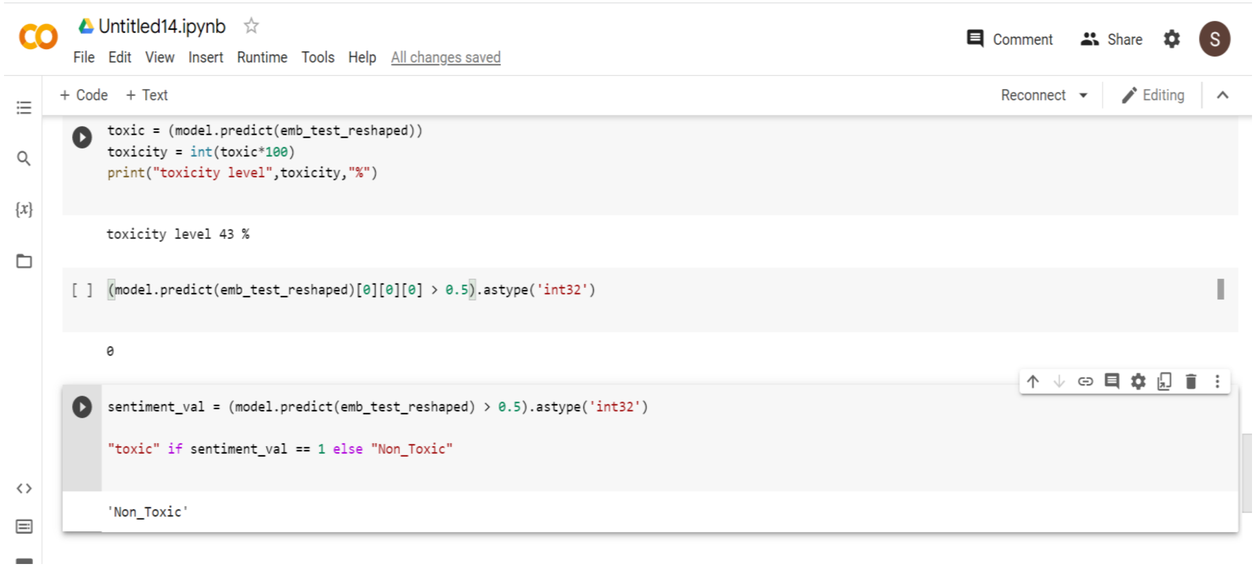
```
TensorShape([1, 1, 512])
```

```
np.squeeze(model.predict(emb_test_reshaped))[()]
```

```
0.43564063
```

```
toxic = (model.predict(emb_test_reshaped))
toxicity = int(toxic*100)
print("toxicity level",toxicity,"%")
```

0s completed at 6:26 PM



```
toxic = (model.predict(emb_test_reshaped))
toxicity = int(toxic*100)
print("toxicity level",toxicity,"%")
```

```
toxicity level 43 %
```

```
(model.predict(emb_test_reshaped)[0][0][0] > 0.5).astype('int32')
```

```
0
```

```
sentiment_val = (model.predict(emb_test_reshaped) > 0.5).astype('int32')

"toxic" if sentiment_val == 1 else "Non_Toxic"
```

```
'Non_Toxic'
```

7. CONCLUSION:

In conclusion, We have developed a comment classification model which classifies the comments using the LSTM model and Universal sentence encoder model, We got an accuracy of 86% .

8. REFERENCES:

[1] H.Almerekhi, H.kwak, j.salmin are these comment triggering? Predicting triggers of toxicity in online discussions,Apr.2020,pp.3033-3040.

[2] M. Anand, R. Eswari, Classification of Abusive Comments in Social Media using Deep Learning, 2019 3rd International Conference on Computing Methodologiesand Communication (ICCMC), Erode, India, Mar. 2019, pp. 974–977.

[3]] A. Bleiweiss, LSTM neural networks for transfer learning in online moderation of abuse context, ICAART 2019 - Proceedings of the 11th International Conference on Agents and Artificial Intelligence, Prague, Czech Republic,2019, pp. 112–122.

[5] E. Brassard-Gourdeau, R. Khoury, Using Sentiment Information for Preemptive Detection of Toxic Comments in Online Conversations, ArXiv200610145 Cs, Jun.2020.

[6] S. Carta, A. Corriga, R. Mulas, D. R. Recupero, R. Saia, A supervised multi-class multi-label word embeddings approach for toxic comment classification, IC3K 2019 - Proceedings of the 11th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management ,vinena,2020.

[7] A. G. D’Sa, I. Illina, D. Fohr, Towards non-toxic landscapes: Automatic toxic comment detection using DNN, ArXiv191108395 Cs Stat, Nov. 2019, Accessed: Jul. 03, 2020.

[8] Toxic comment classification identify and classify toxic online by jigsaw. In 2017.

[9] S. Deshmukh, R. Rade, Tackling Toxic Online Communication with Recurrent Capsule Networks, 2018 Conference on Information and Communication Technology (CICT), Jabalpur, India, 2018.

[10] S. Carta, A. Corriga, R. Mulas, D. R. Recupero, R. Saia, A supervised multi-class multi-label word embeddings approach for toxic comment classification, IC3K 2019 - Proceedings of the 11th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management ,vinena,2020.