```
import tensorflow as tf
import pandas as pd
import numpy as np
```

```
df=pd.read_csv('/content/drive/MyDrive/Comment_Toxicity_Project/train.csv')
df.shape
```

```
(159571, 8)
```

```
df.isna().sum()
```

```
id               0
comment_text     0
toxic            0
severe_toxic     0
obscene          0
threat           0
insult           0
identity_hate    0
dtype: int64
```

```
df.head()
```

	id	comment_text	toxic	severe_toxic	obscene	threat	insult	identity_hate
0	0000997932d777bf	Explanation\nWhy the edits made under my usern...	0	0	0	0	0	0
1	000103f0d9cfb60f	D'aww! He matches this background colour I'm s...	0	0	0	0	0	0
2	000113f07ec002fd	Hey man, I'm really not trying to edit war. It...	0	0	0	0	0	0
4	0001d958c54c6e35	You, sir, are my hero. Any chance you remember	0	0	0	0	0	0

Preprocessing of the dataset

```
from tensorflow.keras.layers import TextVectorization
```

```
x=df['comment_text']
y=df.iloc[:,2:].values
```

```
MAX_FEATURES=100000 #number of words in the vocabulary
vectorizer=TextVectorization(
    max_tokens=MAX_FEATURES,
    output_sequence_length=1800,
    output_mode='int'
)
```

```
vectorizer.adapt(x.values)
```

```
vectorized_text=vectorizer(x.values)
```

```
vectorized_text
```

```
<tf.Tensor: shape=(159571, 1800), dtype=int64, numpy=
array([[  645,    76,     2, ...,     0,     0,     0],
       [    1,    54,  2489, ...,     0,     0,     0],
       [  425,   441,    70, ...,     0,     0,     0],
       ...,
```

```
[32445,  7392,   383, ...,     0,     0,     0],
[    5,    12,   534, ...,     0,     0,     0],
[    5,     8,   130, ...,     0,     0,     0]])>
```

```
dataset=tf.data.Dataset.from_tensor_slices((vectorized_text,y))
dataset=dataset.cache()
dataset=dataset.batch(16)
dataset=dataset.prefetch(buffer_size=tf.data.experimental.AUTOTUNE)
```

```
x_batch,y_batch=dataset.as_numpy_iterator().next()
```

```
train=dataset.take(int(len(dataset)*0.7))
val=dataset.skip(int(len(dataset)*0.7)).take(int(len(dataset)*0.2))
test=dataset.skip(int(len(dataset)*0.9)).take(int(len(dataset)*0.1))
```

```
train_generator= train.as_numpy_iterator()
```

Creating the deep neural network

```
model = tf.keras.Sequential()
from tensorflow.keras.layers import LSTM, Dense ,Bidirectional ,Dropout ,Embedding
```

```
#Create the embedded layer
model.add(Embedding(MAX_FEATURES+1,32))
#Bidirectional LSTM layer
model.add(Bidirectional(LSTM(32,activation='tanh')))
#Feature extracting FULLY CONNECTED LAYER
model.add(Dense(128,activation='relu'))
model.add(Dense(256,activation='relu'))
model.add(Dense(128,activation='relu'))
```

```
#Output layer or output layer
model.add(Dense(6,activation='sigmoid'))
```

```
model.compile(loss='BinaryCrossentropy',optimizer='adam')
```

```
# model.summary()
history = model.fit(train,epochs=1,validation_data=val)
```

```
6981/6981 [==============================] - 876s 124ms/step - loss: 0.0619 - val_loss: 0.0489
```

```
text='I will kill you !'
test_token= vectorizer(text)
```

```
res = model.predict(np.expand_dims(test_token,0))
res
```

```
1/1 [==============================] - 1s 749ms/step
array([[0.7946706 , 0.02397404, 0.41763267, 0.02712157, 0.3702678 ,
        0.05644592]], dtype=float32)
```

```
batch_x,batch_y=test.as_numpy_iterator().next()
```

```
(model.predict(batch_x)>0.5).astype(int)
```

```
1/1 [==============================] - 0s 63ms/step
array([[0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0],
```

```
       [0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0]])
```

Evaluate the model

```
from tensorflow.keras.metrics import Precision,Recall,CategoricalAccuracy
pre=Precision()
re=Recall()
cat=CategoricalAccuracy()
for batch in test.as_numpy_iterator():
  x_true,y_true=batch
  y_hat=model.predict(x_true)
  y_hat=y_hat.flatten()
  y_true=y_true.flatten()
  pre.update_state(y_true,y_hat)
  re.update_state(y_true,y_hat)
  cat.update_state(y_true,y_hat)
```

```
1/1 [==============================] - 0s 56ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 55ms/step
1/1 [==============================] - 0s 54ms/step
1/1 [==============================] - 0s 55ms/step
1/1 [==============================] - 0s 55ms/step
1/1 [==============================] - 0s 56ms/step
```

```
print(f"precision:{pre.result().numpy()},recall:{re.result().numpy()},Categorical Accuracy:{cat.result().numpy()}")
```

```
precision:0.8438966870307922,recall:0.6045403480529785,Categorical Accuracy:0.45235708355903625
```

```
# !pip install gradio jinja2
```

```
import gradio as gd
model.save('toxicity.h5')
```

```
model=tf.keras.models.load_model('toxicity.h5')
(model.predict(np.expand_dims(test_token,1))>0.5).astype(int)
```

```
57/57 [==============================] - 30s 3ms/step
array([[0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0],
       [1, 0, 0, 0, 0, 0],
       ...,
       [0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0]])
```

```
def prediction(comment):
  vectorized=vectorizer([comment])
  result=model.predict(vectorized)
  text=''
  for idx,col in enumerate(df.columns[2:]):
    text+='{}:{},'.format(col, result[0][idx]>0.3)
  return text
```

```
prediction('You are so terrible')
```

```
1/1 [==============================] - 0s 74ms/step
'toxic:False,severe_toxic:False,obscene:False,threat:False,insult:False,identity_hate:False,'
```

```
interface= gd.Interface(fn=prediction,
                        inputs=gd.inputs.Textbox(lines=2,placeholder='Comment to evaluate'),
                                              outputs='text' )
```

```
<ipython-input-66-f80580da2d53>:2: GradioDeprecationWarning: Usage of gradio.inputs is deprecated, and will not be supported in
  inputs=gd.inputs.Textbox(lines=2,placeholder='Comment to evaluate'),
<ipython-input-66-f80580da2d53>:2: GradioDeprecationWarning: `optional` parameter is deprecated, and it has no effect
  inputs=gd.inputs.Textbox(lines=2,placeholder='Comment to evaluate'),
<ipython-input-66-f80580da2d53>:2: GradioDeprecationWarning: `numeric` parameter is deprecated, and it has no effect
  inputs=gd.inputs.Textbox(lines=2,placeholder='Comment to evaluate'),
```

```
interface.launch(share=True)
```

```
Rerunning server... use `close()` to stop if you need to change `launch()` parameters.
----
Colab notebook detected. To show errors in colab notebook, set debug=True in launch()
Running on public URL: https://4e8784ac245fe460e7.gradio.live

This share link expires in 72 hours. For free permanent hosting and GPU upgrades, run `gradio deploy` from Terminal to deploy t
```

comment

Comment to evaluate

Clear

Submit

output

Flag

Use via API · Built with Gradio

✓ 1s completed at 10:31 PM