In [42]:
```
import numpy as np
import pandas as pd

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

In [43]:
```
import tensorflow as tf
import pandas as pd
from sklearn.model_selection import train_test_split
from keras.preprocessing import text,sequence
from keras.utils import pad_sequences
from keras.models import Sequential
from keras.layers import Embedding,SimpleRNN,LSTM,SpatialDropout1D,GRU,Bidirectional,Input,Dense,Activation,Dro
# from keras.layers.core import Dense#,Activation,Dropout
```

In [44]:
```
from tqdm import tqdm
```

In [45]:
```
#configurint TPU
try:
    tpu = tf.distribute.cluster_resolver.TPUClusterResolver()
    print('Running on TPU', tpu.master())
except ValueError:
    tpu = None

if tpu:
    tf.config.experimental_connect_to_cluster(tpu)
    tf.tpu.experimental.initialize_tpu_system(tpu)
    strategy = tf.distribute.experimental.TPUStrategy(tpu)
else:
    #default distribution strategy in tensorflow, Works on CPU and single GPU
    strategy = tf.distribute.OneDeviceStrategy("CPU:0")
```

In [46]:
```
train = pd.read_csv('C:/Users/Richa/OneDrive/Desktop/TARP PROJECT_TOXIC COMMENT/dataset/jigsaw-toxic-comment-tr
validation = pd.read_csv('C:/Users/Richa/OneDrive/Desktop/TARP PROJECT_TOXIC COMMENT/dataset/validation.csv')
test = pd.read_csv('C:/Users/Richa/OneDrive/Desktop/TARP PROJECT_TOXIC COMMENT/dataset/test.csv')
```

In [47]:
```
train.shape
```

Out[47]:
```
(223549, 8)
```

In [48]:
```
train.head()
```

Out[48]:

	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
3	0001b41b1c6bb37e	"\nMore\nI can't make any real suggestions on ...	0	0	0	0	0	0
4	0001d958c54c6e35	You, sir, are my hero. Any chance you remember...	0	0	0	0	0	0

In [49]:
```
train.drop(['severe_toxic','obscene','threat','insult','identity_hate'],axis = 1, inplace = True)
```

In [50]:
```
#check max len of comment_text column to use this for padding in future
pad_len = train['comment_text'].apply(lambda x:len(str(x).split())).max()
print('max len of comment_text column',pad_len)
```

```
max len of comment_text column 2321
```

DATA PREPARATION

In [51]:
```
xtrain, xvalid, ytrain, yvalid = train_test_split(train.comment_text.values, train.toxic.values, stratify = tra
```

In [52]:
```
len(xtrain),len(xvalid)
```

Out[52]:
```
(178839, 44710)
```

Tokenisation and Padding with max len of words in curpus

In [53]:
```
test.head()
```

Out[53]:

	id	content	lang
0	0	Doctor Who adlı viki başlığına 12. doctor olar...	tr
1	1	Вполне возможно, но я пока не вижу необходимо...	ru
2	2	Quindi tu sei uno di quelli conservativi , ...	it
3	3	Malesef gerçekleştirilmedi ancak şöyle bir şey...	tr
4	4	:Resim:Seldabagcan.jpg resminde kaynak sorunu ...	tr

In [54]:
```
#using keras tokenizer
token = text.Tokenizer(num_words = None)
max_len = 2400
xtest = test.content.values
token.fit_on_texts(list(xtrain) + list(xvalid) + list(xtest))

x_train_seq = token.texts_to_sequences(xtrain)
x_valid_seq = token.texts_to_sequences(xvalid)
x_test_seq = token.texts_to_sequences(xtest)

#zero pad the sequences
x_train_pad = pad_sequences(x_train_seq,maxlen = max_len)


x_valid_pad = pad_sequences(x_valid_seq,maxlen = max_len)
x_test_pad = pad_sequences(x_test_seq,maxlen = max_len)
word_index = token.word_index
```

Classification based on GRU(Gated Recurrent Unit)

In [55]:
```
#create an embedding metrics for the words which are part of our datasets
embedding_metrics = np.zeros((len(word_index) + 1,300))
for word,i in tqdm(word_index.items()):
    embedding_vector = embeddings_index.get(word)
    if embedding_vector is not None:
        embedding_metrics[i] = embedding_vector
```

```
100%|██████████| 583776/583776 [00:00<00:00, 1205366.26it/s]
```

In [56]:
```
%%time

with strategy.scope():

    model = Sequential()
    model.add(Embedding(len(word_index) + 1,
                        300,
                        weights = [embedding_metrics],
                        input_length = max_len,
                        trainable = False))
    model.add(SpatialDropout1D(0.3))
    model.add(GRU(300))
    model.add(Dense(1, activation = 'sigmoid'))

    model.compile(loss = 'binary_crossentropy', optimizer = 'adam', metrics = ['accuracy'])
model.summary()
```

```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_2 (Embedding)     (None, 2400, 300)         175133100

 spatial_dropout1d_2 (Spatia  (None, 2400, 300)        0
 lDropout1D)

 gru_2 (GRU)                 (None, 300)               541800

 dense_2 (Dense)             (None, 1)                 301

=================================================================
Total params: 175,675,201
Trainable params: 542,101
Non-trainable params: 175,133,100
_________________________________________________________________
Wall time: 2.36 s
```

In [ ]:
```
model.fit(x_train_pad,ytrain, epochs = 1,batch_size = 128*strategy.num_replicas_in_sync)
```

```
 362/1398 [======>.....................] - ETA: 30:58:28 - loss: 0.3277 - accuracy: 0.9044
```

In [ ]:
```
gru_pred = model.predict(x_valid_pad)
```

In [ ]:
```
model_accuracy = roc_auc_score(yvalid,gru_pred)
model_accuracy_ls.append({'model':'GRU','AUC_SCORE':model_accuracy})
```

In [ ]:
```
model_accuracy_ls
```

In [ ]:
```

```