LSTM

Importing packages

In [1]:

```
# Credits: https://machinelearningmastery.com/sequence-classification-lstm-recurrent-neural
#https://www.liip.ch/en/blog/sentiment-detection-with-keras-word-embeddings-and-lstm-deep-l

# LSTM for sequence classification in the IMDB dataset
import numpy
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import LSTM
from keras.layers.embeddings import Embedding
from tensorflow.keras.preprocessing import text,sequence
from tensorflow.keras.layers import Activation, Add, Bidirectional, Conv1D, Dense, Dropout,
from tensorflow.keras.layers import concatenate, GRU, Input, CuDNNLSTM, MaxPooling1D
from tensorflow.keras.layers import GlobalAveragePooling1D,  GlobalMaxPooling1D, SpatialDro
from tensorflow.keras.layers import MaxPooling1D
from tensorflow.keras.layers import BatchNormalization
from tensorflow.keras.models import Model
from tensorflow.keras.optimizers import Adam, RMSprop
from sklearn.metrics import accuracy_score, roc_auc_score, log_loss
from sklearn.model_selection import train_test_split
from tensorflow.keras import initializers, regularizers, constraints, optimizers, layers, c
from tensorflow.keras.callbacks import Callback, EarlyStopping,ModelCheckpoint
from tensorflow.keras.initializers import he_normal
from tensorflow.keras.regularizers import l2
from tensorflow.python.keras import backend as k

import tensorflow.keras
from tensorflow.keras import optimizers
from tensorflow.keras.initializers import he_normal
from tensorflow.keras.layers import BatchNormalization
from time import time
from tensorflow.python.keras.callbacks import TensorBoard

# fix random seed for reproducibility
numpy.random.seed(7)

import numpy as np
import pandas as pd
from tqdm import tqdm
import warnings
warnings.filterwarnings("ignore")
```

```
Using TensorFlow backend.
```

Loading Data

In [2]:

```
df = pd.read_csv("train_data.csv")
```

In [3]:

```
df.head(3)
```

Out[3]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs.	IN	
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr.	FL	
2	21895	p182444	3465aaf82da834c0582ebd0ef8040ca0	Ms.	AZ	

'Resource_summary_has_digit'

In [4]:

```
import re
def hasNumbers(inputString):
    return bool(re.search(r'\d',inputString))

resource_summary=list(df['project_resource_summary'].values)
has_digits = []
for i in resource_summary:
    if (hasNumbers(i)==True):
        has_digits.append(1)
    else:
        has_digits.append(0)
```

'project_title'

In [5]:

```
# https://stackoverflow.com/a/47091490/4084039
import re


def decontracted(phrase):
    # specific
    phrase = re.sub(r"won't", "will not", phrase)
    phrase = re.sub(r"can\'t", "can not", phrase)
    # general
    phrase = re.sub(r"n\'t", " not", phrase)
    phrase = re.sub(r"\'re", " are", phrase)
    phrase = re.sub(r"\'s", " is", phrase)
    phrase = re.sub(r"\'d", " would", phrase)
    phrase = re.sub(r"\'ll", " will", phrase)
    phrase = re.sub(r"\'t", " not", phrase)
    phrase = re.sub(r"\'ve", " have", phrase)
    phrase = re.sub(r"\'m", " am", phrase)
    return phrase
```

In [6]:

```
# https://gist.github.com/sebleier/554280
# we are removing the words from the stop words list: 'no', 'nor'
stopwords= ['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "
            "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his
            'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'itself', 'they'
            'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', "that'l
            'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had',
            'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'u
            'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through', 'd
            'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over',
            'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', '
            'most', 'other', 'some', 'such', 'only', 'own', 'same', 'so', 'than', 'too', 'v
            's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've", 'now',
            've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "didn't", 'do
            "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn', "isn't", 'ma', 'mightn',
            "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shouldn't", 'wasn'
            'won', "won't", 'wouldn', "wouldn't"]


from tqdm import tqdm
preprocessed_title = []
# tqdm is for printing the status bar
for sentance in tqdm(df['project_title'].values):
    sent = decontracted(sentance)
    sent = sent.replace('\\r', ' ')
    sent = sent.replace('nannan', ' ')
    sent = sent.replace('\\"', ' ')
    sent = sent.replace('\\n', ' ')
    sent = sent.lower()
    sent = re.sub('[^A-Za-z0-9]+', ' ', sent)
    # https://gist.github.com/sebleier/554280
    sent = ' '.join(e for e in sent.split(" ") if e not in stopwords)
    preprocessed_title.append(sent.lower().strip())
```

```
100%|██████████████████████████████████████████████████████████████████████████████
| 109248/109248 [00:02<00:00, 51669.89it/s]
```

Typesetting math: 0%

'project_resource_summary'

In [7]:

```
from tqdm import tqdm
summary = []
# tqdm is for printing the status bar
for sentance in tqdm(df['project_resource_summary'].values):
    sent = decontracted(sentance)
    sent = sent.replace('\\r', ' ')
    sent = sent.replace('nannan', ' ')
    sent = sent.replace('\\"', ' ')
    sent = sent.replace('\\n', ' ')
    sent = sent.lower()
    sent = re.sub('[^A-Za-z0-9]+', ' ', sent)
    # https://gist.github.com/sebleier/554280
    sent = ' '.join(e for e in sent.split(" ") if e not in stopwords)
    summary.append(sent.lower().strip())
```

```
100%|██████████████████████████████████████████████████████████████████████████████
| 109248/109248 [00:04<00:00, 22683.49it/s]
```

In [8]:

```
project_data = pd.read_csv("preprocessed_data.csv")
```

In [9]:

```
project_data["title"] = preprocessed_title
```

In [10]:

```
project_data['summary'] = summary
```

In [11]:

```
project_data['summary_digits']=has_digits
```

In [12]:

```
project_data.shape
```

Out[12]:

```
(109248, 12)
```

In [13]:

```
project_data.head()
```

Out[13]:

	school_state	teacher_prefix	project_grade_category	teacher_number_of_previously_posted_pr
0	ca	mrs	grades_prek_2	
1	ut	ms	grades_3_5	
2	ca	mrs	grades_prek_2	
3	ga	mrs	grades_prek_2	
4	wa	mrs	grades_3_5	

In [14]:

```
#Printing the attributes of project_data
print("Attributes :", project_data.columns.values)
```

```
Attributes : ['school_state' 'teacher_prefix' 'project_grade_category'
 'teacher_number_of_previously_posted_projects' 'project_is_approved'
 'clean_categories' 'clean_subcategories' 'essay' 'price' 'title'
 'summary' 'summary_digits']
```

Resource data

In [15]:

```
resource_data = pd.read_csv('resources.csv')
```

In [16]:

```
resource_data.head(3)
```

Out[16]:

	id	description	quantity	price
0	p233245	LC652 - Lakeshore Double-Space Mobile Drying Rack	1	149.00
1	p069063	Bouncy Bands for Desks (Blue support pipes)	3	14.95
2	p069063	Cory Stories: A Kid's Book About Living With Adhd	1	8.45

In [17]:

```
# reference : https://stackoverflow.com/questions/22407798/how-to-reset-a-dataframes-indexe
price_data = resource_data.groupby('id').agg({'price':'sum', 'quantity':'sum'}).reset_index
price_data.head(2)
```

Out[17]:

	id	price	quantity
0	p000001	459.56	7
1	p000002	515.89	21

In [18]:

```
# join two dataframes(project_data and price_data) in python
# reference : https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.m
project_data['price'] = resource_data['price']
project_data['quantity'] = resource_data['quantity']
```

In [19]:

```
project_data.head(2)
```

Out[19]:

	school_state	teacher_prefix	project_grade_category	teacher_number_of_previously_posted_pr
0	ca	mrs	grades_prek_2	
1	ut	ms	grades_3_5	

'Total numerical features'

Typesetting math: 0%

In [20]:

```
#numerical inputs
project_data['num'] = project_data['teacher_number_of_previously_posted_projects'] + projec
```

'Total text data'

In [21]:

```
project_data['total_text'] = project_data['essay']+ " " +project_data['title']+ " " + proje
```

In [22]:

```
project_data.head()
```

Out[22]:

	school_state	teacher_prefix	project_grade_category	teacher_number_of_previously_posted_pr
0	ca	mrs	grades_prek_2	
1	ut	ms	grades_3_5	
2	ca	mrs	grades_prek_2	
3	ga	mrs	grades_prek_2	
4	wa	mrs	grades_3_5	

In [23]:

```
col = ['teacher_number_of_previously_posted_projects', 'price', 'quantity','essay','title',

project_data.drop(labels=col,axis =1, inplace=True)
```


Final dataset

In [24]:

```
project_data.head()
```

Out[24]:

	school_state	teacher_prefix	project_grade_category	project_is_approved	clean_categories
0	ca	mrs	grades_prek_2	1	math_science
1	ut	ms	grades_3_5	1	specialneeds
2	ca	mrs	grades_prek_2	1	literacy_language
3	ga	mrs	grades_prek_2	1	appliedlearning
4	wa	mrs	grades_3_5	1	literacy_language

In [25]:

```
y = project_data['project_is_approved']
project_data.drop(['project_is_approved'], axis=1, inplace=True)
X = project_data

print(X.shape)
print(y.shape)
```

```
(109248, 7)
(109248,)
```

Preparing Data for model

Splitting data, stratify sampling

In [26]:

```
#splitting data
X_train,X_test,y_train,y_test = train_test_split(X, y , stratify = y, train_size = 0.7)
X_train,X_cv,y_train,y_cv = train_test_split(X_train,y_train,stratify = y_train,train_size
```

In [27]:

```
print(X_train.shape, y_train.shape)
print(X_cv.shape, y_cv.shape)
print(X_test.shape, y_test.shape)
```

```
(53531, 7) (53531,)
(22942, 7) (22942,)
(32775, 7) (32775,)
```

Label to categorical

In [28]:

```
#converting class labels to categorical variables
from keras.utils import to_categorical
y_train = to_categorical(y_train)
y_test = to_categorical(y_test)
y_cv = to_categorical(y_cv)
```

Featurization and padding text data

In [36]:

```
from keras.preprocessing.text import Tokenizer

t = Tokenizer()
t.fit_on_texts(X_train['total_text'])
word_index = t.word_index
text_train = t.texts_to_sequences(X_train['total_text'])
text_test = t.texts_to_sequences(X_test['total_text'])
text_cv = t.texts_to_sequences(X_cv['total_text'])


from keras.preprocessing.sequence import pad_sequences
max_review_length = 300
text_train = pad_sequences(text_train, maxlen=max_review_length)
text_test = pad_sequences(text_test, maxlen=max_review_length)
text_cv = pad_sequences(text_cv, maxlen=max_review_length)
```

In [37]:

```
text_train[10]
```

Out[37]:

```
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,   10, 1572,   70, 2212,
       1567,    7,    1,    5, 1572, 1008, 1541,    7,    1,  621, 1767,
        114,   14,  947, 2425,  279,  327,  264, 2425,  279,  197,  456,
          3,   31,  162,  177,   38,   39,   89,    2,   50,  125,   38,
        284,    3,  524,  455, 4935,    1,  657,   72,  173,   92,   14,
         25,   24, 1564,  326,    7,    1,   19,   24,  484,  968,  603,
        649, 2767,    7,    3, 2505,   24,  215,    1,   56,    2, 4017,
        395,  725,   11,  420,   24,  572, 2682,   49,    3,    7,  162,
         51,    6,   83,    1,  385,   24,  177,   90,  572,  539,   26,
          1,   71,  263,  177, 1947, 1008, 1541,  441, 2064, 2250,   26,
         29, 2177, 2401, 2273,  572,  505,  406,  939,   17,   71,  781,
       3488,   90,  127,    8, 2141, 2059,    1,   71,  150,  177, 1947,
       1008, 1541,  441, 2064, 2250,   26,   29, 2177, 2401, 2273,  572,
        505,  406,  939,   17,   71,  781, 3488,   90,  127,    8, 2141,
       2059,    7,    1,   11, 1714, 1137,   90,    8,  327,  279,  320,
        150,  103,   87,  379,  224, 1133,  424,  371, 2528, 3403, 1753,
        168,  132,    1,  493,  223,    2,  320,   51,  128,    1,   11,
         30,  395,   24,   26,    7,    1,  854, 2340,   24, 2035,   24,
        168,  114,  409,    2,  143, 1384,   27, 7246,  569,    1,  160,
         15, 4155,  678,  764,  427,    1,    1,    4,  764,  427, 4155,
        678, 2327,   36])
```

Creating embedding matrix using pretrain golve model

In [29]:

```
embeddings_index = {}
f = open('glove.42B.300d.txt', encoding="utf8")
for line in f:
    values = line.split()
    word = values[0]
    coefs = np.asarray(values[1:], dtype='float32')
    embeddings_index[word] = coefs
f.close()

print('Found %s word vectors.' % len(embeddings_index))
```

```
Found 1917495 word vectors.
```

In [38]:

```
embedding_matrix = np.zeros((len(word_index) + 1, 300))
for word, i in word_index.items():
    embedding_vector = embeddings_index.get(word)
    if embedding_vector is not None:
        # words not found in embedding index will be all-zeros.
        embedding_matrix[i] = embedding_vector
```

In [39]:

```
embedding_matrix
```

Out[39]:

```
array([[ 0.        ,  0.        ,  0.        , ...,  0.        ,
         0.        ,  0.        ],
       [ 0.15243   , -0.16945   , -0.022748  , ...,  0.61800998,
         0.41281   ,  0.0010077 ],
       [-0.043504  , -0.18483999, -0.14613   , ...,  0.1008    ,
         0.1068    ,  0.089065  ],
       ...,
       [ 0.        ,  0.        ,  0.        , ...,  0.        ,
         0.        ,  0.        ],
       [ 0.        ,  0.        ,  0.        , ...,  0.        ,
         0.        ,  0.        ],
       [ 0.14264999, -0.20883   ,  0.53634   , ...,  0.19422001,
         0.062518  ,  0.018873  ]])
```

Tokenizing categorical data

1. School_state

In [40]:

```
from sklearn import preprocessing

le = preprocessing.LabelEncoder()
le.fit(X_train['school_state'])

state_train = le.transform(X_train['school_state'])
state_test = le.transform(X_test['school_state'])
state_cv = le.transform(X_cv['school_state'])

print(state_train.shape)
```

```
(53531,)
```

2.Teacher_prefix

In [41]:

```
le = preprocessing.LabelEncoder()
le.fit(X_train['teacher_prefix'])

prefix_train = le.transform(X_train['teacher_prefix'])
prefix_test = le.transform(X_test['teacher_prefix'])
prefix_cv = le.transform(X_cv['teacher_prefix'])

print(prefix_train.shape)
```

```
(53531,)
```

3. Project_grade_category

In [42]:

```
le = preprocessing.LabelEncoder()
le.fit(X_train['project_grade_category'])

grade_train = le.transform(X_train['project_grade_category'])
grade_test = le.transform(X_test['project_grade_category'])
grade_cv = le.transform(X_cv['project_grade_category'])

print(grade_train.shape)
```

```
(53531,)
```

4. Clean_categories

In [43]:

```
t = Tokenizer()
t.fit_on_texts(X_train['clean_categories'])
word_index = t.word_index
clean_cat_train = t.texts_to_sequences(X_train['clean_categories'])
clean_cat_test = t.texts_to_sequences(X_test['clean_categories'])
clean_cat_cv = t.texts_to_sequences(X_cv['clean_categories'])

from keras.preprocessing.sequence import pad_sequences
max_review_length = 5
clean_cat_train = pad_sequences(clean_cat_train, maxlen=max_review_length)
clean_cat_test = pad_sequences(clean_cat_test, maxlen=max_review_length)
clean_cat_cv = pad_sequences(clean_cat_cv, maxlen=max_review_length)

print(clean_cat_train.shape)
```

```
(53531, 5)
```

5. Clean_subcategories

In [44]:

```
t = Tokenizer()
t.fit_on_texts(X_train['clean_subcategories'])
word_index = t.word_index
clean_subcat_train = t.texts_to_sequences(X_train['clean_subcategories'])
clean_subcat_test = t.texts_to_sequences(X_test['clean_subcategories'])
clean_subcat_cv = t.texts_to_sequences(X_cv['clean_subcategories'])

from keras.preprocessing.sequence import pad_sequences
max_review_length = 5
clean_subcat_train = pad_sequences(clean_subcat_train, maxlen=max_review_length)
clean_subcat_test = pad_sequences(clean_subcat_test, maxlen=max_review_length)
clean_subcat_cv = pad_sequences(clean_subcat_cv, maxlen=max_review_length)

print(clean_subcat_train.shape)
```

```
(53531, 5)
```

Numerical feature

In [45]:

```
from sklearn.preprocessing import StandardScaler
scalar = StandardScaler()
scalar.fit(X_train['num'].values.reshape(-1,1))
# finding the mean and standar

# Now standardize the data with above maen and variance.
num_train = scalar.transform(X_train['num'].values.reshape(-1, 1))
num_cv = scalar.transform(X_cv['num'].values.reshape(-1, 1))
num_test = scalar.transform(X_test['num'].values.reshape(-1, 1))

print(num_train.shape)
```

```
(53531, 1)
```

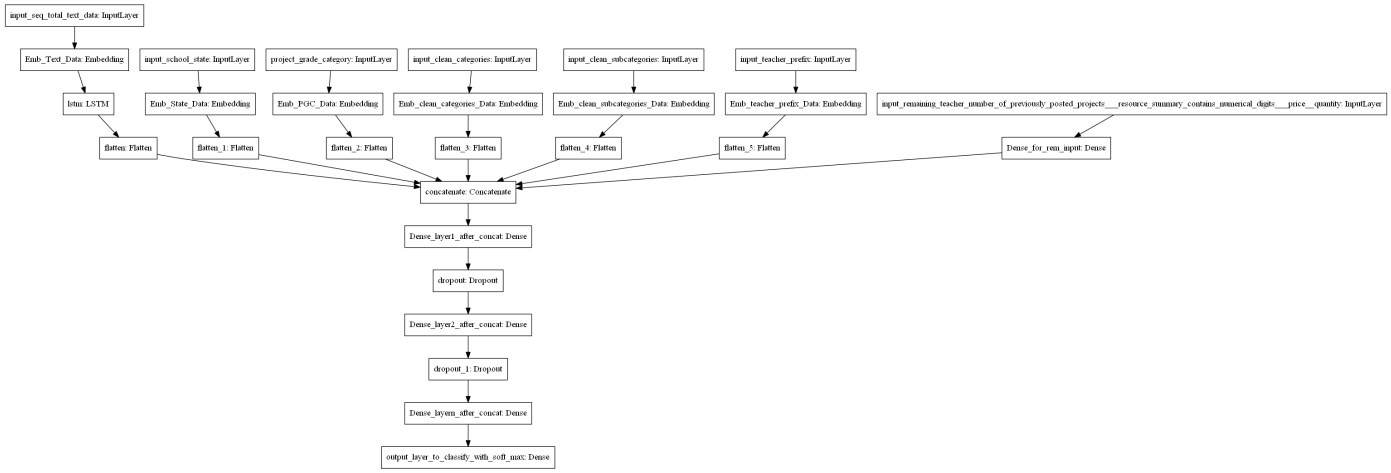
Function for AUC Score

In [46]:

```
#https://stackoverflow.com/questions/41032551/how-to-compute-receiving-operating-characteri
#https://developpaper.com/question/how-to-apply-the-custom-operation-of-py_func-in-tensorfl
def auc(y_true, y_pred) :
    score = tf.py_func( lambda y_true, y_pred : roc_auc_score(y_true, y_pred, average='macr
                        [y_true, y_pred],
                        'float32',
                        stateful=True,
                        name='sklearnAUC')
    return score
```

LSTM Model 1

Typesetting math: 0%



ref: https://i.imgur.com/w395Yk9.png

To find the embedding size and vocab

Typesetting math: 0%

In [47]:

```
no_of_unique_cat  = X_train['school_state'].nunique()
embedding_size = min(np.ceil((no_of_unique_cat)/2), 50 )
print("School_state")
print("embedding_output_dim",int(embedding_size))
print('vocab', no_of_unique_cat+1)
print("-"*50)

no_of_unique_cat  = X_train['teacher_prefix'].nunique()
embedding_size = min(np.ceil((no_of_unique_cat)/2), 50 )
print("Teacher_prefix")
print("embedding_output_dim",int(embedding_size))
print('vocab', no_of_unique_cat+1)
print("-"*50)

no_of_unique_cat  = X_train['project_grade_category'].nunique()
embedding_size = min(np.ceil((no_of_unique_cat)/2), 50 )
print("Grade_category")
print("embedding_output_dim",int(embedding_size))
print('vocab', no_of_unique_cat+1)
print("-"*50)

no_of_unique_cat  = X_train['clean_categories'].nunique()
embedding_size = min(np.ceil((no_of_unique_cat)/2), 50 )
print("Subject_categories")
print("embedding_output_dim",int(embedding_size))
print('vocab', no_of_unique_cat+1)
print("-"*50)

no_of_unique_cat  = X_train['clean_subcategories'].nunique()
embedding_size = min(np.ceil((no_of_unique_cat)/2), 50 )
print("Subject_categories")
print("embedding_output_dim",int(embedding_size))
print('vocab', no_of_unique_cat+1)
```

```
School_state
embedding_output_dim 26
vocab 52
--------------------------------------------------
Teacher_prefix
embedding_output_dim 3
vocab 6
--------------------------------------------------
Grade_category
embedding_output_dim 2
vocab 5
--------------------------------------------------
Subject_categories
embedding_output_dim 25
vocab 51
--------------------------------------------------
Subject_categories
embedding_output_dim 50
vocab 376
```

In [48]:

```
tf.keras.backend.clear_session()

#Essay input --> 1
text = Input(shape=(300,), name="text")
x1 = Embedding(input_dim=46454,output_dim=300,trainable=False,weights=[embedding_matrix])(t
x1 = SpatialDropout1D(0.3)(x1)
x1 = CuDNNLSTM(256,return_sequences=True)(x1)
x1 = Flatten()(x1)


#State input --> 2
state = Input(shape=(1,), name="state")
x2 = Embedding(input_dim=52,output_dim=26)(state)
x2 = Flatten()(x2)

#Teacher prefix input --> 3
prefix = Input(shape=(1,), name="prefix")
x3 = Embedding(input_dim=6,output_dim=3)(prefix)
x3 = Flatten()(x3)

#Grade category input --> 4
grade = Input(shape=(1,), name="grade")
x4 = Embedding(input_dim=5,output_dim=2)(grade)
x4 = Flatten()(x4)

#Subject category input --> 5
subj_cat = Input(shape=(5,), name="subject_category")
x5 = Embedding(input_dim=51,output_dim=25)(subj_cat)
x5 = Flatten()(x5)

#Subject subcategory input --> 6
subj_subcat = Input(shape=(5,), name="subject_sub_category")
x6 = Embedding(input_dim=376,output_dim=50)(subj_subcat)
x6 = Flatten()(x6)


#Numerical input -->7
num = Input(shape=(1,), name="numerical")
x7 = (Dense(16, activation='relu'))(num)



concat = concatenate([x1,x2,x3,x4,x5,x6,x7])

x = Dense(128,activation='relu',kernel_initializer=he_normal(),kernel_regularizer=l2(0.0001
x = Dropout(0.4)(x)
x = Dense(64,activation='relu',kernel_initializer=he_normal(),kernel_regularizer=l2(0.0001)
x = Dropout(0.3)(x)
x = BatchNormalization()(x)
x = Dense(32,activation='relu',kernel_initializer=he_normal(),kernel_regularizer=l2(0.0001)

output = Dense(2, activation = 'softmax')(x)


model = Model([text,state,prefix,grade,subj_cat,subj_subcat,num], output)

#https://www.youtube.com/watch?v=2U6Jl7oqRkM
#Installing tensorboard for model visualization
#To visualize, run -  tensorboard --log_dir=logs/{} in command prompt
```

Typesetting math: 0%

```
tensorboard = TensorBoard(log_dir="logs/".format(time))

model.compile(loss="categorical_crossentropy", optimizer= tensorflow.keras.optimizers.Adam(

print(model.summary())
```

```
WARNING:tensorflow:From C:\Users\vansh\Anaconda3\envs\env\lib\site-packages
\tensorflow_core\python\keras\initializers.py:119: calling RandomUniform.__i
nit__ (from tensorflow.python.ops.init_ops) with dtype is deprecated and wil
l be removed in a future version.
Instructions for updating:
Call initializer instance with the dtype argument instead of passing it to t
he constructor
WARNING:tensorflow:From C:\Users\vansh\Anaconda3\envs\env\lib\site-packages
\tensorflow_core\python\ops\resource_variable_ops.py:1630: calling BaseResou
rceVariable.__init__ (from tensorflow.python.ops.resource_variable_ops) with
constraint is deprecated and will be removed in a future version.
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
WARNING:tensorflow:From <ipython-input-46-9c0679feb436>:8: py_func (from ten
sorflow.python.ops.script_ops) is deprecated and will be removed in a future
version.
Instructions for updating:
tf.py_func is deprecated in TF V2. Instead, there are two
    options available in V2.
    - tf.py_function takes a python function which manipulates tf eager
    tensors instead of numpy arrays. It's easy to convert a tf eager tensor
to
    an ndarray (just call tensor.numpy()) but having access to eager tensors
    means `tf.py_function`s can use accelerators such as GPUs as well as
    being differentiable using a gradient tape.
    - tf.numpy_function maintains the semantics of the deprecated tf.py_func
    (it is not differentiable, and manipulates numpy arrays). It drops the
    stateful argument making all functions stateful.

Model: "model"
__________________________________________________________________________
_____________________
Layer (type)                 Output Shape              Param #     Connected t
o
==========================================================================
======================
text (InputLayer)            [(None, 300)]             0
__________________________________________________________________________
_____________________
embedding (Embedding)        (None, 300, 300)          13936200    text[0][0]
__________________________________________________________________________
_____________________
spatial_dropout1d (SpatialDropo (None, 300, 300)       0           embedding
[0][0]
__________________________________________________________________________
_____________________
state (InputLayer)           [(None, 1)]               0
__________________________________________________________________________
_____________________
prefix (InputLayer)          [(None, 1)]               0
__________________________________________________________________________
_____________________
grade (InputLayer)           [(None, 1)]               0
__________________________________________________________________________
```

```
__________________________
subject_category (InputLayer)   [(None, 5)]          0
_________________________________________________________________
__________________________
subject_sub_category (InputLaye [(None, 5)]          0
_________________________________________________________________
__________________________
cu_dnnlstm (CuDNNLSTM)          (None, 300, 256)     571392      spatial_dro
pout1d[0][0]
_________________________________________________________________
__________________________
embedding_1 (Embedding)         (None, 1, 26)        1352        state[0][0]
_________________________________________________________________
__________________________
embedding_2 (Embedding)         (None, 1, 3)         18          prefix[0]
[0]
_________________________________________________________________
__________________________
embedding_3 (Embedding)         (None, 1, 2)         10          grade[0][0]
_________________________________________________________________
__________________________
embedding_4 (Embedding)         (None, 5, 25)        1275        subject_cat
egory[0][0]
_________________________________________________________________
__________________________
embedding_5 (Embedding)         (None, 5, 50)        18800       subject_sub
_category[0][0]
_________________________________________________________________
__________________________
numerical (InputLayer)          [(None, 1)]          0
_________________________________________________________________
__________________________
flatten (Flatten)               (None, 76800)        0           cu_dnnlstm
[0][0]
_________________________________________________________________
__________________________
flatten_1 (Flatten)             (None, 26)           0           embedding_1
[0][0]
_________________________________________________________________
__________________________
flatten_2 (Flatten)             (None, 3)            0           embedding_2
[0][0]
_________________________________________________________________
__________________________
flatten_3 (Flatten)             (None, 2)            0           embedding_3
[0][0]
_________________________________________________________________
__________________________
flatten_4 (Flatten)             (None, 125)          0           embedding_4
[0][0]
_________________________________________________________________
__________________________
flatten_5 (Flatten)             (None, 250)          0           embedding_5
[0][0]
_________________________________________________________________
__________________________
dense (Dense)                   (None, 16)           32          numerical
[0][0]
_________________________________________________________________
__________________________
concatenate (Concatenate)       (None, 77222)        0           flatten[0]
```

```
[0]                                                                 flatten_1
[0][0]
                                                                    flatten_2
[0][0]
                                                                    flatten_3
[0][0]
                                                                    flatten_4
[0][0]
                                                                    flatten_5
[0][0]
                                                                    dense[0][0]
__________________________________________________________________________________
_____________________
dense_1 (Dense)                 (None, 128)          9884544     concatenate
[0][0]
__________________________________________________________________________________
_____________________
dropout (Dropout)               (None, 128)          0           dense_1[0]
[0]
__________________________________________________________________________________
_____________________
dense_2 (Dense)                 (None, 64)           8256        dropout[0]
[0]
__________________________________________________________________________________
_____________________
dropout_1 (Dropout)             (None, 64)           0           dense_2[0]
[0]
__________________________________________________________________________________
_____________________
batch_normalization (BatchNorma (None, 64)           256         dropout_1
[0][0]
__________________________________________________________________________________
_____________________
dense_3 (Dense)                 (None, 32)           2080        batch_norma
lization[0][0]
__________________________________________________________________________________
_____________________
dense_4 (Dense)                 (None, 2)            66          dense_3[0]
[0]
==================================================================================
======================
Total params: 24,424,281
Trainable params: 10,487,953
Non-trainable params: 13,936,328
__________________________________________________________________________________
_____________________
None
```

In [49]:

```
filepath="weights_1.hdf5"
checkpoint = ModelCheckpoint(filepath, monitor='val_auc', verbose=1, save_best_only=True, m

model.fit([text_train,state_train,grade_train,prefix_train,clean_cat_train,clean_subcat_tra
```

```
Train on 53531 samples, validate on 22942 samples
Epoch 1/15
53400/53531 [============================>.] - ETA: 0s - loss: 0.5248 - auc:
0.5486
Epoch 00001: val_auc improved from -inf to 0.39083, saving model to weights_
1.hdf5
53531/53531 [==============================] - 100s 2ms/sample - loss: 0.524
7 - auc: 0.5488 - val_loss: 0.5428 - val_auc: 0.3908
Epoch 2/15
53400/53531 [============================>.] - ETA: 0s - loss: 0.4518 - auc:
0.6576  E - ETA: 16s - lo - ETA: 14s - loss: 0.4583 - auc:  - ETA: 13s - los
s: 0.4591 - a
Epoch 00002: val_auc improved from 0.39083 to 0.70053, saving model to weigh
ts_1.hdf5
53531/53531 [==============================] - 43s 812us/sample - loss: 0.45
18 - auc: 0.6571 - val_loss: 0.4320 - val_auc: 0.7005
Epoch 3/15
53400/53531 [============================>.] - ETA: 0s - loss: 0.4320 - auc:
0.6872
Epoch 00003: val_auc improved from 0.70053 to 0.71539, saving model to weigh
ts_1.hdf5
53531/53531 [==============================] - 43s 805us/sample - loss: 0.43
20 - auc: 0.6876 - val_loss: 0.4246 - val_auc: 0.7154
Epoch 4/15
53400/53531 [============================>.] - ETA: 0s - loss: 0.4185 - auc:
0.7096
Epoch 00004: val_auc improved from 0.71539 to 0.72842, saving model to weigh
ts_1.hdf5
53531/53531 [==============================] - 43s 808us/sample - loss: 0.41
85 - auc: 0.7097 - val_loss: 0.4108 - val_auc: 0.7284
Epoch 5/15
53400/53531 [============================>.] - ETA: 0s - loss: 0.4114 - auc:
0.7195
Epoch 00005: val_auc improved from 0.72842 to 0.73348, saving model to weigh
ts_1.hdf5
53531/53531 [==============================] - 43s 812us/sample - loss: 0.41
13 - auc: 0.7197 - val_loss: 0.4126 - val_auc: 0.7335
Epoch 6/15
53400/53531 [============================>.] - ETA: 0s - loss: 0.4039 - auc:
0.7365
Epoch 00006: val_auc did not improve from 0.73348
53531/53531 [==============================] - 42s 777us/sample - loss: 0.40
41 - auc: 0.7368 - val_loss: 0.4042 - val_auc: 0.7329
Epoch 7/15
53400/53531 [============================>.] - ETA: 0s - loss: 0.3987 - auc:
0.7430
Epoch 00007: val_auc improved from 0.73348 to 0.73648, saving model to weigh
ts_1.hdf5
53531/53531 [==============================] - 44s 816us/sample - loss: 0.39
88 - auc: 0.7428 - val_loss: 0.4051 - val_auc: 0.7365
Epoch 8/15
53400/53531 [============================>.] - ETA: 0s - loss: 0.3953 - auc:
```

```
0.7540
Epoch 00008: val_auc improved from 0.73648 to 0.74610, saving model to weigh
ts_1.hdf5
53531/53531 [==============================] - 43s 812us/sample - loss: 0.39
55 - auc: 0.7537 - val_loss: 0.4068 - val_auc: 0.7461
Epoch 9/15
53400/53531 [============================>.] - ETA: 0s - loss: 0.3910 - auc:
0.7621
Epoch 00009: val_auc did not improve from 0.74610
53531/53531 [==============================] - 42s 778us/sample - loss: 0.39
09 - auc: 0.7626 - val_loss: 0.4064 - val_auc: 0.7456
Epoch 10/15
53400/53531 [============================>.] - ETA: 0s - loss: 0.3910 - auc:
0.7682
Epoch 00010: val_auc did not improve from 0.74610
53531/53531 [==============================] - 42s 779us/sample - loss: 0.39
10 - auc: 0.7675 - val_loss: 0.4125 - val_auc: 0.7430
Epoch 11/15
53400/53531 [============================>.] - ETA: 0s - loss: 0.3909 - auc:
0.7787- ETA: 1s - loss: 0.3909 - auc:
Epoch 00011: val_auc did not improve from 0.74610
53531/53531 [==============================] - 42s 779us/sample - loss: 0.39
09 - auc: 0.7787 - val_loss: 0.4147 - val_auc: 0.7427
Epoch 12/15
53400/53531 [============================>.] - ETA: 0s - loss: 0.3894 - auc:
0.7916
Epoch 00012: val_auc did not improve from 0.74610
53531/53531 [==============================] - 42s 780us/sample - loss: 0.38
93 - auc: 0.7916 - val_loss: 0.4308 - val_auc: 0.7410
Epoch 13/15
53400/53531 [============================>.] - ETA: 0s - loss: 0.3911 - auc:
0.8066
Epoch 00013: val_auc did not improve from 0.74610
53531/53531 [==============================] - 42s 780us/sample - loss: 0.39
10 - auc: 0.8069 - val_loss: 0.4487 - val_auc: 0.7301
Epoch 14/15
53400/53531 [============================>.] - ETA: 0s - loss: 0.3897 - auc:
0.8235
Epoch 00014: val_auc did not improve from 0.74610
53531/53531 [==============================] - 42s 782us/sample - loss: 0.38
96 - auc: 0.8234 - val_loss: 0.4688 - val_auc: 0.7268
Epoch 15/15
53400/53531 [============================>.] - ETA: 0s - loss: 0.3858 - auc:
0.8479
Epoch 00015: val_auc did not improve from 0.74610
53531/53531 [==============================] - 42s 785us/sample - loss: 0.38
59 - auc: 0.8480 - val_loss: 0.5109 - val_auc: 0.7284
```

Out[49]:

```
<tensorflow.python.keras.callbacks.History at 0x1776f005198>
```

compiling model weights

In [50]:

```
model.load_weights("weights_1.hdf5")
model.compile(loss="categorical_crossentropy", optimizer='adam', metrics=[auc])
```

Model visualization

In [51]:

```
print("Train AUC",roc_auc_score(y_train,(model.predict([text_train,state_train,grade_train,
print("-"*50)
print("CV AUC",roc_auc_score(y_cv,(model.predict([text_cv,state_cv,grade_cv,prefix_cv,clean
print("-"*50)
print("Test AUC",roc_auc_score(y_test,(model.predict([text_test,state_test,grade_test,prefi
```

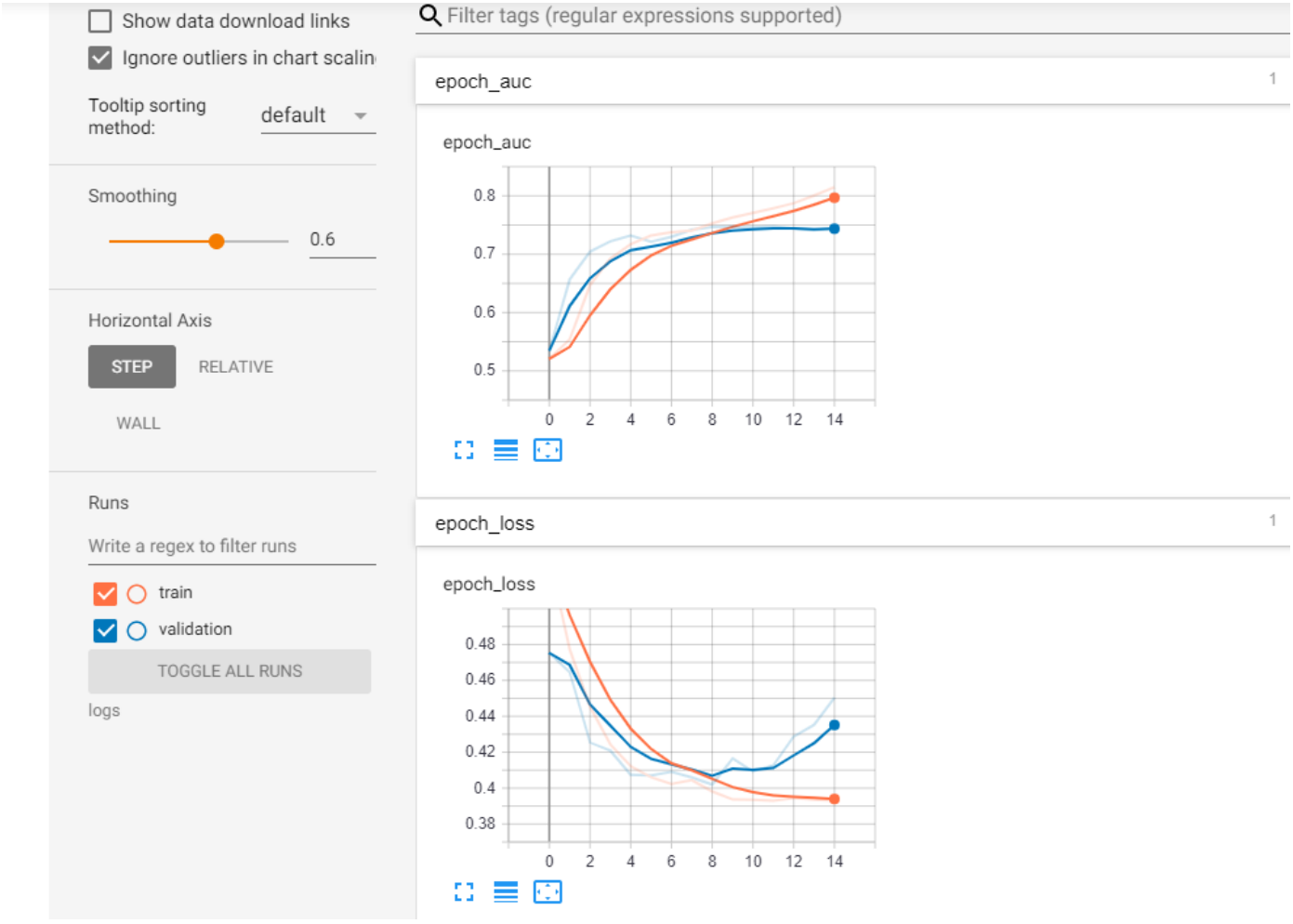
```
Train AUC 0.7864668722630237
--------------------------------------------------
CV AUC 0.7450655336380656
--------------------------------------------------
Test AUC 0.760104170038495
```

In [52]:

```
"""# Load TENSORBOARD
%load_ext tensorboard
# Start TENSORBOARD
%tensorboard --logdir logs --port=8008"""
```

Out[52]:

```
'# Load TENSORBOARD\n%load_ext tensorboard\n# Start TENSORBOARD\n%tensorboar
d --logdir logs --port=8008'
```



Typesetting math: 0%

---------- Model 2 ----------

In [53]:

```
print(X_train.shape, y_train.shape)
print(X_cv.shape, y_cv.shape)
print(X_test.shape, y_test.shape)
```

```
(53531, 7) (53531, 2)
(22942, 7) (22942, 2)
(32775, 7) (32775, 2)
```

In [54]:

```
from sklearn.feature_extraction.text import TfidfVectorizer
import matplotlib.pyplot as plt
tfidf = TfidfVectorizer()
data_text = tfidf.fit_transform(X_train['total_text'])
plt.boxplot(tfidf.idf_)
plt.ylabel("IDF score")
```

Out[54]:

```
Text(0, 0.5, 'IDF score')
```

In [55]:

```
print("25 percentile (idf):", np.percentile(tfidf.idf_,[25]))
print("50 percentile (idf):",np.percentile(tfidf.idf_,[50]))
print("75 percentile (idf):",np.percentile(tfidf.idf_,[75]))
print("90 percentile (idf):",np.percentile(tfidf.idf_,[90]))
print("95 percentile (idf):",np.percentile(tfidf.idf_,[95]))
print("99 percentile (idf):",np.percentile(tfidf.idf_,[99]))
```

```
25 percentile (idf): [9.17998468]
50 percentile (idf): [10.7894226]
75 percentile (idf): [11.1948877]
90 percentile (idf): [11.1948877]
95 percentile (idf): [11.1948877]
99 percentile (idf): [11.1948877]
```

In [56]:

```
feature_idf = zip(tfidf.get_feature_names(),tfidf.idf_)

feature_name = []
for x,y in feature_idf:

    if y >=2 and 11.19:
        feature_name.append(x)
    else:
        pass
```

In [57]:

```
len(feature_name)
```

Out[57]:

```
46385
```

Considering only those features with idf value between 25th and 75th percentile in 'project_essay'

In [61]:

```
def reduced_text(df):
    processed_text = []
    for text in df:
        sent = " "
        words = text.split()
        for word in words:
            if word in feature_name:
                sent = sent +" " + word
            else:
                pass

        processed_text.append(sent)
    return processed_text

train_text_reduced = reduced_text(X_train['total_text'])
test_text_reduced = reduced_text(X_test['total_text'])
cv_text_reduced = reduced_text(X_cv['total_text'])
```

In [62]:

```
train_text_reduced[0]
```

Out[62]:

```
'  privilege working amazing although majority low socioeconomic families br
ing curiosity eagerness everyday diverse backgrounds bring unique perspectiv
e individual education plans unfortunately sedentary structures create obsta
cles best frequent movement options traditional furniture meet needs stress
unable move needed receiving instruction cause disengaged often end receivin
g disciplinary referrals acting movement needs requesting exercise ball chai
rs give alternative traditional chairs offered exercise ball pretty popular
problem ball stability necessary produce quality project special needs senso
ry challenges experience without discomfort traditional seating these chairs
give opportunity exercise ball way keep focused rolling away listening liter
ature books cds well listening center utilize cds levels esl'
```

Featurization and padding text data

In [66]:

```
from keras.preprocessing.text import Tokenizer

t = Tokenizer()
t.fit_on_texts(train_text_reduced)
word_index = t.word_index
text_train_reduced = t.texts_to_sequences(train_text_reduced)
text_test_reduced = t.texts_to_sequences(test_text_reduced)
text_cv_reduced = t.texts_to_sequences(cv_text_reduced)


from keras.preprocessing.sequence import pad_sequences
max_review_length = 200
text_train_reduced = pad_sequences(text_train_reduced, maxlen=max_review_length)
text_test_reduced = pad_sequences(text_test_reduced, maxlen=max_review_length)
text_cv_reduced = pad_sequences(text_cv_reduced, maxlen=max_review_length)
```

In [67]:

```
text_train_reduced[0]
```

Out[67]:

```
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, 1521,   48,  136,  559,  310,   63,  413,   84,  201,  811,
       1711,  189,   83,  122,  201,  261, 2314,  273,   27, 1414,  524,
       4220, 1578,   18,  839,   28, 2820,  185,  197,  527, 1205,  180,
         23, 1066,  949,   72,  183,  696,  159, 1770, 7098,  126,  416,
        696, 6428, 8904, 3588,  185,   23,  335,  462,  471,  103,   43,
        449,  527,  103, 1913,  462,  471, 2337, 1599,  204,  471,  624,
        323, 1157,  336,   20,   73,   23,  314,   99,   65,  161, 6429,
        527,   42,    8,  103,   43,   45,  462,  471,   25,   74,  306,
       2389,  589,  282,  363,    1, 2826,   35,  282,  151,  601, 2826,
        186, 1159])
```

Creating embedding matrix using pretrain golve model

In [68]:

```
embedding_matrix = np.zeros((len(word_index) + 1, 300))
for word, i in word_index.items():
    embedding_vector = embeddings_index.get(word)
    if embedding_vector is not None:
        # words not found in embedding index will be all-zeros.
        embedding_matrix[i] = embedding_vector
```

In [69]:

```
embedding_matrix
```

Out[69]:

```
array([[ 0.        ,  0.        ,  0.        , ...,  0.        ,
         0.        ,  0.        ],
       [ 0.50835001,  0.17623   , -0.16424   , ...,  0.77085   ,
         0.15795   , -0.080663  ],
       [-0.26078001, -0.36897999, -0.022831  , ...,  0.23384   ,
         0.24267   ,  0.091846  ],
       ...,
       [ 0.        ,  0.        ,  0.        , ...,  0.        ,
         0.        ,  0.        ],
       [ 0.        ,  0.        ,  0.        , ...,  0.        ,
         0.        ,  0.        ],
       [ 0.14264999, -0.20883   ,  0.53634   , ...,  0.19422001,
         0.062518  ,  0.018873  ]])
```

In [71]:

```
tf.keras.backend.clear_session()

#Essay input --> 1
text = Input(shape=(200,), name="text")
x1 = Embedding(input_dim=46386,output_dim=300,trainable=False,weights=[embedding_matrix])(t
x1 = SpatialDropout1D(0.3)(x1)
x1 = CuDNNLSTM(256,return_sequences=True)(x1)
x1 = Flatten()(x1)


#State input --> 2
state = Input(shape=(1,), name="state")
x2 = Embedding(input_dim=52,output_dim=26)(state)
x2 = Flatten()(x2)

#Teacher prefix input --> 3
prefix = Input(shape=(1,), name="prefix")
x3 = Embedding(input_dim=6,output_dim=3)(prefix)
x3 = Flatten()(x3)

#Grade category input --> 4
grade = Input(shape=(1,), name="grade")
x4 = Embedding(input_dim=5,output_dim=2)(grade)
x4 = Flatten()(x4)

#Subject category input --> 5
subj_cat = Input(shape=(5,), name="subject_category")
x5 = Embedding(input_dim=51,output_dim=25)(subj_cat)
x5 = Flatten()(x5)

#Subject subcategory input --> 6
subj_subcat = Input(shape=(5,), name="subject_sub_category")
x6 = Embedding(input_dim=376,output_dim=50)(subj_subcat)
x6 = Flatten()(x6)


#Numerical input -->7
num = Input(shape=(1,), name="numerical")
x7 = (Dense(16, activation='relu'))(num)



concat = concatenate([x1,x2,x3,x4,x5,x6,x7])

x = Dense(128,activation='relu',kernel_initializer=he_normal(),kernel_regularizer=l2(0.0001
x = Dropout(0.4)(x)
x = Dense(64,activation='relu',kernel_initializer=he_normal(),kernel_regularizer=l2(0.0001)
x = Dropout(0.3)(x)
x = BatchNormalization()(x)
x = Dense(32,activation='relu',kernel_initializer=he_normal(),kernel_regularizer=l2(0.0001)

output = Dense(2, activation = 'softmax')(x)


model = Model([text,state,prefix,grade,subj_cat,subj_subcat,num], output)

#https://www.youtube.com/watch?v=2U6Jl7oqRkM
#Installing tensorboard for model visualization
#To visualize, run -  tensorboard --log_dir=logs/{} in command prompt
```

Typesetting math: 0%

```
tensorboard = TensorBoard(log_dir="logs/".format(time))

model.compile(loss="categorical_crossentropy", optimizer= tensorflow.keras.optimizers.Adam(

print(model.summary())
```

```
Model: "model"
_____________________________________________________________________________
_____________________
Layer (type)                 Output Shape              Param #     Connected
to
=============================================================================
=========================
text (InputLayer)            [(None, 200)]             0
_____________________________________________________________________________
_____________________
embedding (Embedding)        (None, 200, 300)          13915800    text[0]
[0]
_____________________________________________________________________________
_____________________
spatial_dropout1d (SpatialDropo (None, 200, 300)       0           embedding
[0][0]
_____________________________________________________________________________
_____________________
state (InputLayer)           [(None, 1)]               0
_____________________________________________________________________________
_____________________
prefix (InputLayer)          [(None, 1)]               0
_____________________________________________________________________________
_____________________
grade (InputLayer)           [(None, 1)]               0
_____________________________________________________________________________
_____________________
subject_category (InputLayer)   [(None, 5)]            0
_____________________________________________________________________________
_____________________
subject_sub_category (InputLaye [(None, 5)]            0
_____________________________________________________________________________
_____________________
cu_dnnlstm (CuDNNLSTM)       (None, 200, 256)          571392      spatial_d
ropout1d[0][0]
_____________________________________________________________________________
_____________________
embedding_1 (Embedding)      (None, 1, 26)             1352        state[0]
[0]
_____________________________________________________________________________
_____________________
embedding_2 (Embedding)      (None, 1, 3)              18          prefix[0]
[0]
_____________________________________________________________________________
_____________________
embedding_3 (Embedding)      (None, 1, 2)              10          grade[0]
[0]
_____________________________________________________________________________
_____________________
embedding_4 (Embedding)      (None, 5, 25)             1275        subject_c
ategory[0][0]
_____________________________________________________________________________
_____________________
```

```
embedding_5 (Embedding)         (None, 5, 50)        18800       subject_s
ub_category[0][0]
__________________________________________________________________________
_____________________
numerical (InputLayer)          [(None, 1)]          0
__________________________________________________________________________
_____________________
flatten (Flatten)               (None, 51200)        0           cu_dnnlst
m[0][0]
__________________________________________________________________________
_____________________
flatten_1 (Flatten)             (None, 26)           0           embedding
_1[0][0]
__________________________________________________________________________
_____________________
flatten_2 (Flatten)             (None, 3)            0           embedding
_2[0][0]
__________________________________________________________________________
_____________________
flatten_3 (Flatten)             (None, 2)            0           embedding
_3[0][0]
__________________________________________________________________________
_____________________
flatten_4 (Flatten)             (None, 125)          0           embedding
_4[0][0]
__________________________________________________________________________
_____________________
flatten_5 (Flatten)             (None, 250)          0           embedding
_5[0][0]
__________________________________________________________________________
_____________________
dense (Dense)                   (None, 16)           32          numerical
[0][0]
__________________________________________________________________________
_____________________
concatenate (Concatenate)       (None, 51622)        0           flatten
[0][0]
                                                                 flatten_1
[0][0]
                                                                 flatten_2
[0][0]
                                                                 flatten_3
[0][0]
                                                                 flatten_4
[0][0]
                                                                 flatten_5
[0][0]
                                                                 dense[0]
[0]
__________________________________________________________________________
_____________________
dense_1 (Dense)                 (None, 128)          6607744     concatena
te[0][0]
__________________________________________________________________________
_____________________
dropout (Dropout)               (None, 128)          0           dense_1
[0][0]
__________________________________________________________________________
_____________________
dense_2 (Dense)                 (None, 64)           8256        dropout
[0][0]
```

```
_________________________________________________________________
dropout_1 (Dropout)          (None, 64)                0         dense_2
[0][0]
_________________________________________________________________
batch_normalization (BatchNorma (None, 64)                256       dropout_1
[0][0]
_________________________________________________________________
dense_3 (Dense)              (None, 32)                2080      batch_nor
malization[0][0]
_________________________________________________________________
dense_4 (Dense)              (None, 2)                 66        dense_3
[0][0]
==================================================================
Total params: 21,127,081
Trainable params: 7,211,153
Non-trainable params: 13,915,928
_________________________________________________________________
None
```

In [72]:

```
filepath="weights_2.hdf5"
checkpoint = ModelCheckpoint(filepath, monitor='val_auc', verbose=1, save_best_only=True, m

model.fit([text_train_reduced,state_train,grade_train,prefix_train,clean_cat_train,clean_su
```

```
Train on 53531 samples, validate on 22942 samples
Epoch 1/15
53500/53531 [============================>.] - ETA: 0s - loss: 0.5560 - auc:
0.5364
Epoch 00001: val_auc improved from -inf to 0.62902, saving model to weights_
2.hdf5
53531/53531 [==============================] - 38s 714us/sample - loss: 0.55
61 - auc: 0.5370 - val_loss: 0.4657 - val_auc: 0.6290
Epoch 2/15
53500/53531 [============================>.] - ETA: 0s - loss: 0.4785 - auc:
0.6116
Epoch 00002: val_auc improved from 0.62902 to 0.67708, saving model to weigh
ts_2.hdf5
53531/53531 [==============================] - 29s 537us/sample - loss: 0.47
85 - auc: 0.6124 - val_loss: 0.4454 - val_auc: 0.6771
Epoch 3/15
53500/53531 [============================>.] - ETA: 0s - loss: 0.4515 - auc:
0.6698
Epoch 00003: val_auc improved from 0.67708 to 0.67890, saving model to weigh
ts_2.hdf5
53531/53531 [==============================] - 29s 536us/sample - loss: 0.45
15 - auc: 0.6690 - val_loss: 0.4354 - val_auc: 0.6789
Epoch 4/15
53500/53531 [============================>.] - ETA: 0s - loss: 0.4398 - auc:
0.6843
Epoch 00004: val_auc improved from 0.67890 to 0.68632, saving model to weigh
ts_2.hdf5
53531/53531 [==============================] - 29s 536us/sample - loss: 0.43
97 - auc: 0.6841 - val_loss: 0.4279 - val_auc: 0.6863
Epoch 5/15
53500/53531 [============================>.] - ETA: 0s - loss: 0.4272 - auc:
0.7012
Epoch 00005: val_auc improved from 0.68632 to 0.71009, saving model to weigh
ts_2.hdf5
53531/53531 [==============================] - 28s 531us/sample - loss: 0.42
72 - auc: 0.7012 - val_loss: 0.4209 - val_auc: 0.7101
Epoch 6/15
53500/53531 [============================>.] - ETA: 0s - loss: 0.4182 - auc:
0.7131
Epoch 00006: val_auc improved from 0.71009 to 0.71548, saving model to weigh
ts_2.hdf5
53531/53531 [==============================] - 29s 542us/sample - loss: 0.41
82 - auc: 0.7116 - val_loss: 0.4132 - val_auc: 0.7155
Epoch 7/15
53500/53531 [============================>.] - ETA: 0s - loss: 0.4125 - auc:
0.7239
Epoch 00007: val_auc improved from 0.71548 to 0.72226, saving model to weigh
ts_2.hdf5
53531/53531 [==============================] - 29s 533us/sample - loss: 0.41
24 - auc: 0.7245 - val_loss: 0.4099 - val_auc: 0.7223
Epoch 8/15
53500/53531 [============================>.] - ETA: 0s - loss: 0.4073 - auc:
0.7287
```

```
Epoch 00008: val_auc improved from 0.72226 to 0.72404, saving model to weigh
ts_2.hdf5
53531/53531 [==============================] - 29s 542us/sample - loss: 0.40
73 - auc: 0.7291 - val_loss: 0.4099 - val_auc: 0.7240
Epoch 9/15
53500/53531 [============================>.] - ETA: 0s - loss: 0.4022 - auc:
0.7418
Epoch 00009: val_auc improved from 0.72404 to 0.72722, saving model to weigh
ts_2.hdf5
53531/53531 [==============================] - 29s 541us/sample - loss: 0.40
22 - auc: 0.7423 - val_loss: 0.4096 - val_auc: 0.7272
Epoch 10/15
53500/53531 [============================>.] - ETA: 0s - loss: 0.4002 - auc:
0.7502
Epoch 00010: val_auc improved from 0.72722 to 0.73366, saving model to weigh
ts_2.hdf5
53531/53531 [==============================] - 29s 545us/sample - loss: 0.40
01 - auc: 0.7509 - val_loss: 0.4099 - val_auc: 0.7337
Epoch 11/15
53500/53531 [============================>.] - ETA: 0s - loss: 0.3981 - auc:
0.7566
Epoch 00011: val_auc did not improve from 0.73366
53531/53531 [==============================] - 28s 518us/sample - loss: 0.39
82 - auc: 0.7558 - val_loss: 0.4145 - val_auc: 0.7303
Epoch 12/15
53500/53531 [============================>.] - ETA: 0s - loss: 0.3973 - auc:
0.7660
Epoch 00012: val_auc improved from 0.73366 to 0.73417, saving model to weigh
ts_2.hdf5
53531/53531 [==============================] - 29s 537us/sample - loss: 0.39
74 - auc: 0.7660 - val_loss: 0.4179 - val_auc: 0.7342
Epoch 13/15
53500/53531 [============================>.] - ETA: 0s - loss: 0.3958 - auc:
0.7780
Epoch 00013: val_auc improved from 0.73417 to 0.73569, saving model to weigh
ts_2.hdf5
53531/53531 [==============================] - 29s 542us/sample - loss: 0.39
58 - auc: 0.7785 - val_loss: 0.4233 - val_auc: 0.7357
Epoch 14/15
53500/53531 [============================>.] - ETA: 0s - loss: 0.3923 - auc:
0.7969
Epoch 00014: val_auc did not improve from 0.73569
53531/53531 [==============================] - 28s 519us/sample - loss: 0.39
23 - auc: 0.7966 - val_loss: 0.4348 - val_auc: 0.7306
Epoch 15/15
53500/53531 [============================>.] - ETA: 0s - loss: 0.3912 - auc:
0.8123
Epoch 00015: val_auc did not improve from 0.73569
53531/53531 [==============================] - 28s 519us/sample - loss: 0.39
13 - auc: 0.8126 - val_loss: 0.4574 - val_auc: 0.7260
```

Out[72]:

```
<tensorflow.python.keras.callbacks.History at 0x17891485a90>
```

compiling model weights

In [73]:

```
model.load_weights("weights_2.hdf5")
model.compile(loss="categorical_crossentropy", optimizer='adam', metrics=[auc])
```

Model visualization

In [74]:

```
print("Train AUC",roc_auc_score(y_train,(model.predict([text_train_reduced,state_train,grad
print("-"*50)
print("CV AUC",roc_auc_score(y_cv,(model.predict([text_cv_reduced,state_cv,grade_cv,prefix_
print("-"*50)
print("Test AUC",roc_auc_score(y_test,(model.predict([text_test_reduced,state_test,grade_te
```

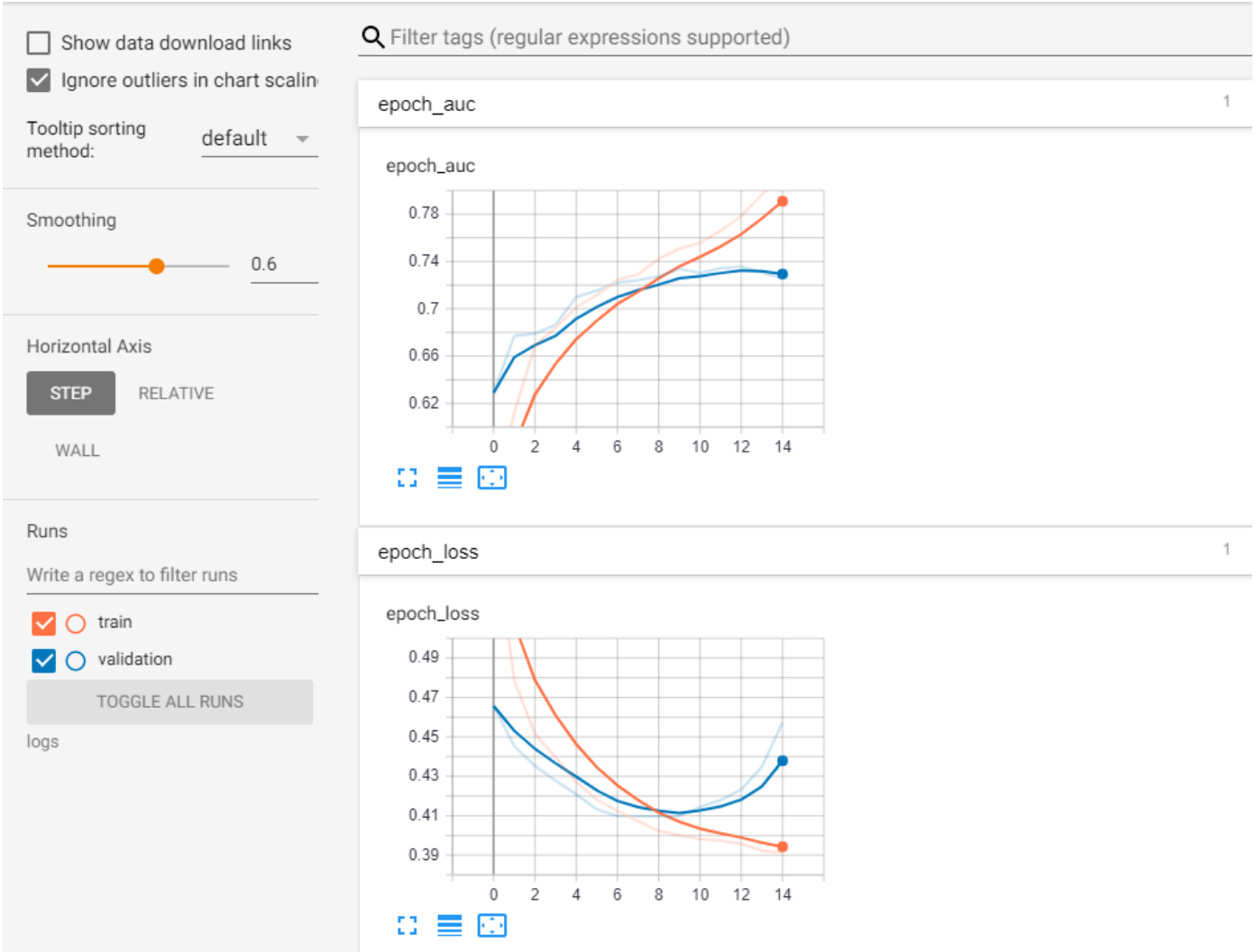
```
Train AUC 0.8240843559440708
--------------------------------------------------
CV AUC 0.7360593751179179
--------------------------------------------------
Test AUC 0.7463178622047653
```

In [77]:

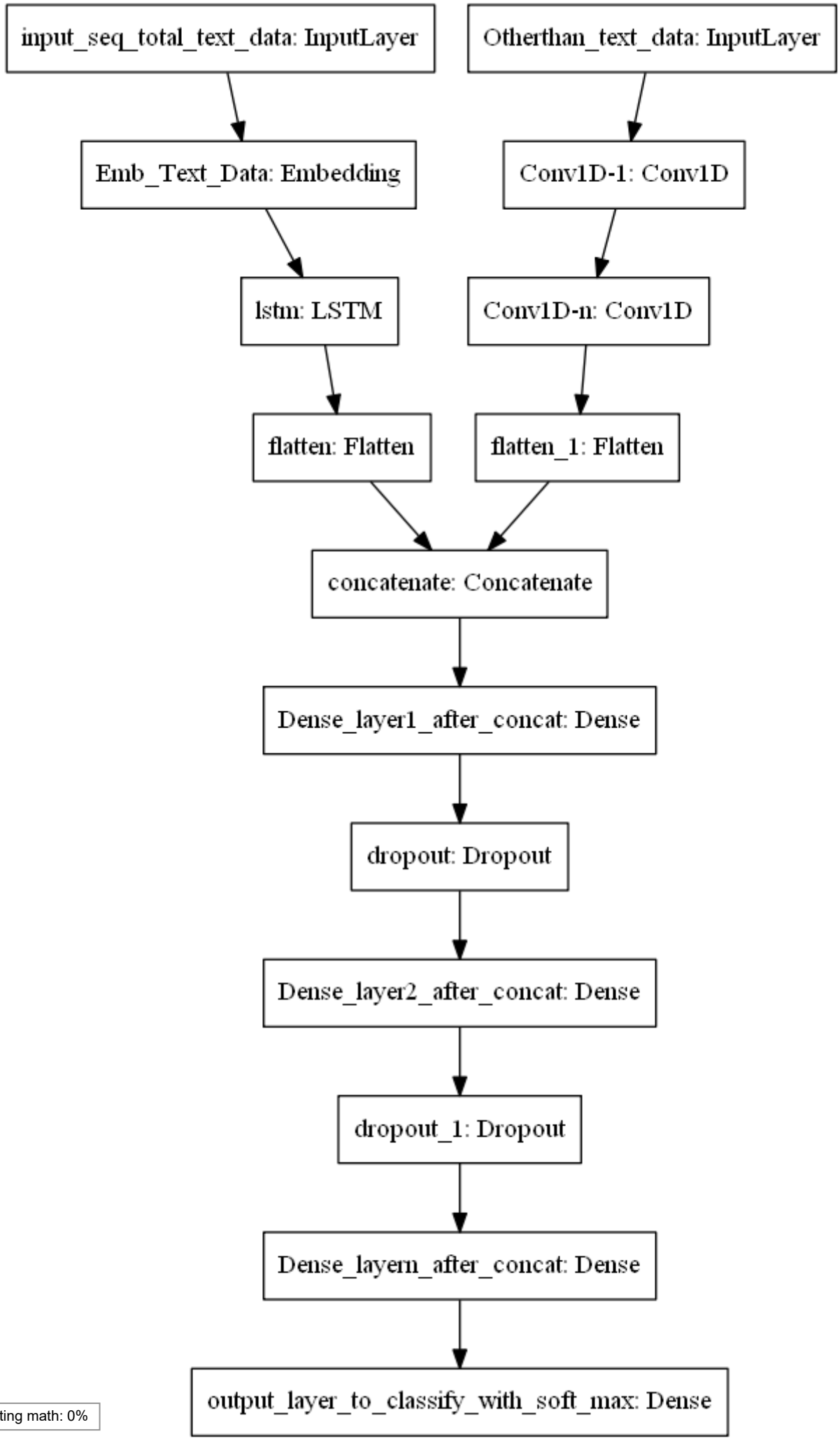
```
"""# Load TENSORBOARD
%load_ext tensorboard
# Start TENSORBOARD
%tensorboard --logdir logs --port=8008"""
```

Out[77]:

```
'# Load TENSORBOARD\n%load_ext tensorboard\n# Start TENSORBOARD\n%tensorboar
d --logdir logs --port=8008'
```



---------- Model 3 ----------



ref: https://i.imgur.com/fkQ8nGo.png

In [35]:

```
from sklearn.feature_extraction.text import CountVectorizer
count = CountVectorizer(lowercase= False)
token = CountVectorizer()

school_state_train = (token.fit_transform(X_train['school_state'])).toarray()
school_state_test = (token.transform(X_test['school_state'])).toarray()
school_state_cv = (token.transform(X_cv['school_state'])).toarray()

print(school_state_train.shape)
```

```
(53531, 51)
```

In [36]:

```
prefix_train = token.fit_transform(X_train['teacher_prefix']).toarray()
prefix_cv = token.transform(X_cv['teacher_prefix']).toarray()
prefix_test = token.transform(X_test['teacher_prefix']).toarray()
print(prefix_train.shape)
```

```
(53531, 5)
```

In [37]:

```
grade_train = token.fit_transform(X_train['project_grade_category']).toarray()
grade_cv = token.transform(X_cv['project_grade_category']).toarray()
grade_test = token.transform(X_test['project_grade_category']).toarray()
print(grade_train.shape)
```

```
(53531, 4)
```

In [38]:

```
cat_train = token.fit_transform(X_train['clean_categories']).toarray()
cat_cv = token.transform(X_cv['clean_categories']).toarray()
cat_test = token.transform(X_test['clean_categories']).toarray()
print(cat_train.shape)
```

```
(53531, 9)
```

In [39]:

```
subcat_train = token.fit_transform(X_train['clean_subcategories']).toarray()
subcat_cv = token.transform(X_cv['clean_subcategories']).toarray()
subcat_test = token.transform(X_test['clean_subcategories']).toarray()
print(subcat_train.shape)
```

```
(53531, 30)
```

In [40]:

```
train_num = X_train['num'].values.reshape(-1,1)
cv_num = X_cv['num'].values.reshape(-1,1)
test_num = X_test['num'].values.reshape(-1,1)
print(train_num.shape)
print(cv_num.shape)
print(test_num.shape)
```

```
(53531, 1)
(22942, 1)
(32775, 1)
```

In [43]:

```
cat_num_train_feat = np.hstack((school_state_train,prefix_train,grade_train,cat_train,subca
cat_num_cv_feat = np.hstack((school_state_cv,prefix_cv,grade_cv,cat_cv,subcat_cv,cv_num))
cat_num_test_feat = np.hstack((school_state_test,prefix_test,grade_test,cat_test,subcat_tes
print(cat_num_train_feat.shape)
print(cat_num_cv_feat.shape)
print(cat_num_test_feat.shape)
```

```
(53531, 100)
(22942, 100)
(32775, 100)
```

In [44]:

```
cat_num_train_feat = np.resize(cat_num_train_feat, new_shape=(53531,100,1))
cat_num_cv_feat = np.resize(cat_num_cv_feat, new_shape=(22942,100,1))
cat_num_test_feat = np.resize(cat_num_test_feat, new_shape=(32775,100,1))
```

In [45]:

```
tf.keras.backend.clear_session()

# input 1
essay = Input(batch_shape=(None,300), name="essay_input")
x1 = Embedding(input_dim=46454,output_dim = 300,weights=[embedding_matrix],trainable = Fals
x1 = SpatialDropout1D(0.4)(x1)
x1 = CuDNNLSTM(256,return_sequences=True)(x1)
x1 = Flatten()(x1)

# input 2
other = Input(shape=(100,1),name="other_input")
x2 = Conv1D(filters=64,kernel_size=3,strides=1)(other)
x2 = BatchNormalization()(x2)
x2 = Conv1D(filters=64,kernel_size=3,strides=1)(x2)
x2 = Flatten()(x2)


concat = concatenate([x1,x2])


x = Dense(64,activation='relu',kernel_initializer=he_normal(),kernel_regularizer=l2(0.0001)
x = Dropout(0.3)(x)
x = Dense(64,activation='relu',kernel_initializer=he_normal(),kernel_regularizer=l2(0.0001)
x = Dropout(0.3)(x)
x = BatchNormalization()(x)
x = Dense(126,activation='relu',kernel_initializer=he_normal(),kernel_regularizer=l2(0.0001

output = Dense(2, activation = 'softmax')(x)

model = Model([essay,other], output)

#To visualize, run -  tensorboard --log_dir=logs/ in command prompt

tensorboard = TensorBoard(log_dir="logs/".format(time))

model.compile(loss="categorical_crossentropy", optimizer= tensorflow.keras.optimizers.Adam(
print(model.summary())
```

```
WARNING:tensorflow:From C:\Users\vansh\Anaconda3\envs\env\lib\site-package
s\tensorflow_core\python\keras\initializers.py:119: calling RandomUniform.
__init__ (from tensorflow.python.ops.init_ops) with dtype is deprecated an
d will be removed in a future version.
Instructions for updating:
Call initializer instance with the dtype argument instead of passing it to
the constructor
WARNING:tensorflow:From C:\Users\vansh\Anaconda3\envs\env\lib\site-package
s\tensorflow_core\python\ops\resource_variable_ops.py:1630: calling BaseRe
sourceVariable.__init__ (from tensorflow.python.ops.resource_variable_ops)
with constraint is deprecated and will be removed in a future version.
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
WARNING:tensorflow:From <ipython-input-34-9c0679feb436>:8: py_func (from t
ensorflow.python.ops.script_ops) is deprecated and will be removed in a fu
ture version.
Instructions for updating:
tf.py_func is deprecated in TF V2. Instead, there are two
    options available in V2.
```

```
    - tf.py_function takes a python function which manipulates tf eager
    tensors instead of numpy arrays. It's easy to convert a tf eager tenso
r to
    an ndarray (just call tensor.numpy()) but having access to eager tenso
rs
    means `tf.py_function`s can use accelerators such as GPUs as well as
    being differentiable using a gradient tape.
    - tf.numpy_function maintains the semantics of the deprecated tf.py_fu
nc
    (it is not differentiable, and manipulates numpy arrays). It drops the
    stateful argument making all functions stateful.

Model: "model"
__________________________________________________________________________
_____
Layer (type)                    Output Shape         Param #     Connected
to
==========================================================================
=====
essay_input (InputLayer)        [(None, 300)]        0
__________________________________________________________________________
_____
other_input (InputLayer)        [(None, 100, 1)]     0
__________________________________________________________________________
_____
embedding (Embedding)           (None, 300, 300)     13936200    essay_inp
ut[0][0]
__________________________________________________________________________
_____
conv1d (Conv1D)                 (None, 98, 64)       256         other_inp
ut[0][0]
__________________________________________________________________________
_____
spatial_dropout1d (SpatialDropo (None, 300, 300)     0           embedding
[0][0]
__________________________________________________________________________
_____
batch_normalization (BatchNorma (None, 98, 64)       256         conv1d[0]
[0]
__________________________________________________________________________
_____
cu_dnnlstm (CuDNNLSTM)          (None, 300, 256)     571392      spatial_d
ropout1d[0][0]
__________________________________________________________________________
_____
conv1d_1 (Conv1D)               (None, 96, 64)       12352       batch_nor
malization[0][0]
__________________________________________________________________________
_____
flatten (Flatten)               (None, 76800)        0           cu_dnnlst
m[0][0]
__________________________________________________________________________
_____
flatten_1 (Flatten)             (None, 6144)         0           conv1d_1
[0][0]
__________________________________________________________________________
_____
concatenate (Concatenate)       (None, 82944)        0           flatten
[0][0]
                                                                 flatten_1
[0][0]
```

```
_________________________________________________________________
_________________________
dense (Dense)                (None, 64)            5308480     concatena
te[0][0]
_________________________________________________________________
_________________________
dropout (Dropout)            (None, 64)            0           dense[0]
[0]
_________________________________________________________________
_________________________
dense_1 (Dense)              (None, 64)            4160        dropout
[0][0]
_________________________________________________________________
_________________________
dropout_1 (Dropout)          (None, 64)            0           dense_1
[0][0]
_________________________________________________________________
_________________________
batch_normalization_1 (BatchNor (None, 64)            256         dropout_1
[0][0]
_________________________________________________________________
_________________________
dense_2 (Dense)              (None, 126)           8190        batch_nor
malization_1[0][0]
_________________________________________________________________
_________________________
dense_3 (Dense)              (None, 2)             254         dense_2
[0][0]
==================================================================
========================
Total params: 19,841,796
Trainable params: 5,905,340
Non-trainable params: 13,936,456
_________________________________________________________________
_________________________
None
```

In [46]:

```
filepath="weights_3.hdf5"
checkpoint = ModelCheckpoint(filepath, monitor='val_auc', verbose=1, save_best_only=True, m

model.fit([text_train,cat_num_train_feat], y_train, epochs=20, verbose=1, batch_size=250, v
```

```
Train on 53531 samples, validate on 22942 samples
Epoch 1/20
53500/53531 [============================>.] - ETA: 0s - loss: 0.5175 - auc:
0.5172
Epoch 00001: val_auc improved from -inf to 0.55825, saving model to weights_
3.hdf5
53531/53531 [==============================] - 140s 3ms/sample - loss: 0.517
6 - auc: 0.5165 - val_loss: 0.4840 - val_auc: 0.5583
Epoch 2/20
53500/53531 [============================>.] - ETA: 0s - loss: 0.4858 - auc:
0.5234
Epoch 00002: val_auc did not improve from 0.55825
53531/53531 [==============================] - 40s 747us/sample - loss: 0.48
57 - auc: 0.5225 - val_loss: 0.4858 - val_auc: 0.5080
Epoch 3/20
53500/53531 [============================>.] - ETA: 0s - loss: 0.4657 - auc:
0.5816
Epoch 00003: val_auc improved from 0.55825 to 0.63197, saving model to weigh
ts_3.hdf5
53531/53531 [==============================] - 41s 768us/sample - loss: 0.46
56 - auc: 0.5826 - val_loss: 0.5069 - val_auc: 0.6320
Epoch 4/20
53500/53531 [============================>.] - ETA: 0s - loss: 0.4401 - auc:
0.6689
Epoch 00004: val_auc improved from 0.63197 to 0.70634, saving model to weigh
ts_3.hdf5
53531/53531 [==============================] - 41s 770us/sample - loss: 0.44
01 - auc: 0.6674 - val_loss: 0.4867 - val_auc: 0.7063
Epoch 5/20
53500/53531 [============================>.] - ETA: 0s - loss: 0.4280 - auc:
0.6933
Epoch 00005: val_auc improved from 0.70634 to 0.71743, saving model to weigh
ts_3.hdf5
53531/53531 [==============================] - 41s 773us/sample - loss: 0.42
79 - auc: 0.6944 - val_loss: 0.4360 - val_auc: 0.7174
Epoch 6/20
53500/53531 [============================>.] - ETA: 0s - loss: 0.4161 - auc:
0.7136
Epoch 00006: val_auc improved from 0.71743 to 0.72785, saving model to weigh
ts_3.hdf5
53531/53531 [==============================] - 42s 776us/sample - loss: 0.41
64 - auc: 0.7142 - val_loss: 0.4666 - val_auc: 0.7278
Epoch 7/20
53500/53531 [============================>.] - ETA: 0s - loss: 0.4078 - auc:
0.7276
Epoch 00007: val_auc improved from 0.72785 to 0.73120, saving model to weigh
ts_3.hdf5
53531/53531 [==============================] - 41s 772us/sample - loss: 0.40
78 - auc: 0.7279 - val_loss: 0.4419 - val_auc: 0.7312
Epoch 8/20
53500/53531 [============================>.] - ETA: 0s - loss: 0.4019 - auc:
0.7375
Epoch 00008: val_auc improved from 0.73120 to 0.73642, saving model to weigh
```

```
ts_3.hdf5
53531/53531 [==============================] - 41s 775us/sample - loss: 0.40
20 - auc: 0.7367 - val_loss: 0.4664 - val_auc: 0.7364
Epoch 9/20
53500/53531 [============================>.] - ETA: 0s - loss: 0.4002 - auc:
0.7410
Epoch 00009: val_auc improved from 0.73642 to 0.74336, saving model to weigh
ts_3.hdf5
53531/53531 [==============================] - 41s 774us/sample - loss: 0.40
01 - auc: 0.7412 - val_loss: 0.4515 - val_auc: 0.7434
Epoch 10/20
53500/53531 [============================>.] - ETA: 0s - loss: 0.3930 - auc:
0.7519
Epoch 00010: val_auc improved from 0.74336 to 0.74618, saving model to weigh
ts_3.hdf5
53531/53531 [==============================] - 42s 776us/sample - loss: 0.39
31 - auc: 0.7512 - val_loss: 0.4804 - val_auc: 0.7462
Epoch 11/20
53500/53531 [============================>.] - ETA: 0s - loss: 0.3907 - auc:
0.7598
Epoch 00011: val_auc improved from 0.74618 to 0.74923, saving model to weigh
ts_3.hdf5
53531/53531 [==============================] - 41s 773us/sample - loss: 0.39
07 - auc: 0.7598 - val_loss: 0.4468 - val_auc: 0.7492
Epoch 12/20
53500/53531 [============================>.] - ETA: 0s - loss: 0.3862 - auc:
0.7673
Epoch 00012: val_auc did not improve from 0.74923
53531/53531 [==============================] - 41s 757us/sample - loss: 0.38
61 - auc: 0.7676 - val_loss: 0.4147 - val_auc: 0.7450
Epoch 13/20
53500/53531 [============================>.] - ETA: 0s - loss: 0.3849 - auc:
0.7709
Epoch 00013: val_auc did not improve from 0.74923
53531/53531 [==============================] - 41s 758us/sample - loss: 0.38
48 - auc: 0.7716 - val_loss: 0.4395 - val_auc: 0.7476
Epoch 14/20
53500/53531 [============================>.] - ETA: 0s - loss: 0.3795 - auc:
0.7804
Epoch 00014: val_auc improved from 0.74923 to 0.75180, saving model to weigh
ts_3.hdf5
53531/53531 [==============================] - 42s 782us/sample - loss: 0.37
95 - auc: 0.7807 - val_loss: 0.4459 - val_auc: 0.7518
Epoch 15/20
53500/53531 [============================>.] - ETA: 0s - loss: 0.3773 - auc:
0.7878
Epoch 00015: val_auc did not improve from 0.75180
53531/53531 [==============================] - 41s 758us/sample - loss: 0.37
73 - auc: 0.7881 - val_loss: 0.4310 - val_auc: 0.7488
Epoch 16/20
53500/53531 [============================>.] - ETA: 0s - loss: 0.3732 - auc:
0.8001
Epoch 00016: val_auc did not improve from 0.75180
53531/53531 [==============================] - 41s 759us/sample - loss: 0.37
33 - auc: 0.7997 - val_loss: 0.4249 - val_auc: 0.7451
Epoch 17/20
53500/53531 [============================>.] - ETA: 0s - loss: 0.3677 - auc:
0.8131
Epoch 00017: val_auc did not improve from 0.75180
53531/53531 [==============================] - 41s 759us/sample - loss: 0.36
78 - auc: 0.8133 - val_loss: 0.4567 - val_auc: 0.7444
```

```
Epoch 18/20
53500/53531 [============================>.] - ETA: 0s - loss: 0.3598 - auc:
0.8314
Epoch 00018: val_auc did not improve from 0.75180
53531/53531 [==============================] - 41s 767us/sample - loss: 0.35
99 - auc: 0.8309 - val_loss: 0.4331 - val_auc: 0.7336
Epoch 19/20
53500/53531 [============================>.] - ETA: 0s - loss: 0.3491 - auc:
0.8511
Epoch 00019: val_auc did not improve from 0.75180
53531/53531 [==============================] - 43s 794us/sample - loss: 0.34
91 - auc: 0.8506 - val_loss: 0.4598 - val_auc: 0.7353
Epoch 20/20
53500/53531 [============================>.] - ETA: 0s - loss: 0.3404 - auc:
0.8703
Epoch 00020: val_auc did not improve from 0.75180
53531/53531 [==============================] - 42s 783us/sample - loss: 0.34
04 - auc: 0.8706 - val_loss: 0.5032 - val_auc: 0.7253
```

Out[46]:

```
<tensorflow.python.keras.callbacks.History at 0x24c0ca27b70>
```

compiling model weights

In [47]:

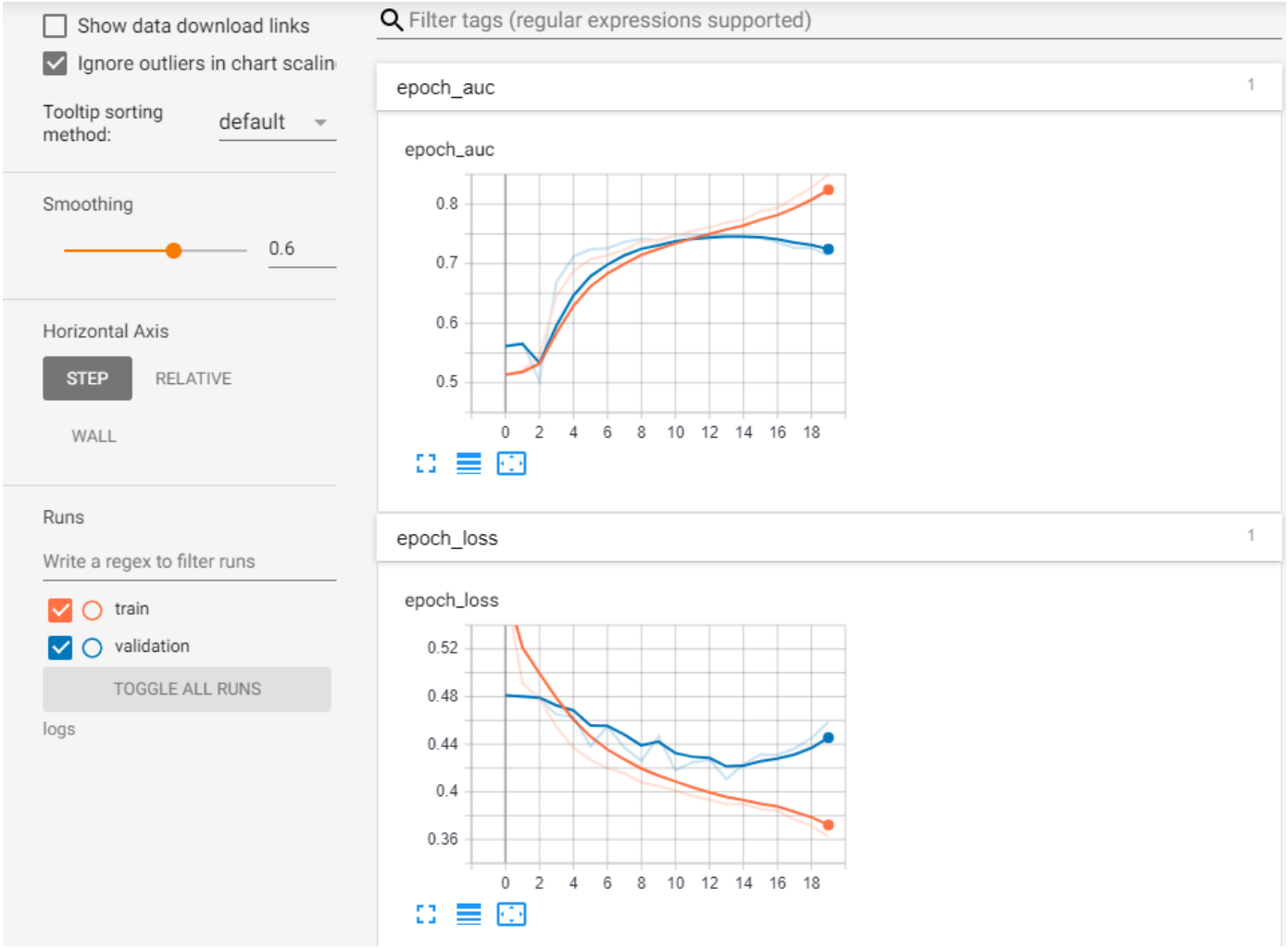
```
from tensorflow.keras.optimizers import Adam
model.load_weights("weights_3.hdf5")
model.compile(loss="categorical_crossentropy", optimizer='adam', metrics=[auc])
```

Model visualization

In [48]:

```
print("Train AUC",roc_auc_score(y_train,(model.predict([text_train,cat_num_train_feat]))))
print("-"*50)
print("Cv AUC",roc_auc_score(y_cv,model.predict([text_cv,cat_num_cv_feat])))
print("-"*50)
print("Test AUC",roc_auc_score(y_test,model.predict([text_test,cat_num_test_feat])))
```

```
Train AUC 0.8204224648514182
--------------------------------------------------
Cv AUC 0.751834165899868
--------------------------------------------------
Test AUC 0.7627339352050855
```



In [76]:

```
from prettytable import PrettyTable

x = PrettyTable(["Model", "Train AUC", "Cv AUC", "Test AUC"])

x.add_row(["Model 1", 0.78,0.74,0.76])
x.add_row(["Model 2", 0.82,0.74,0.75])
x.add_row(["Model 3", 0.82,0.75,0.76])

print(x.get_string(title="Model results"))
```

```
+---------+-----------+--------+----------+
|  Model  | Train AUC | Cv AUC | Test AUC |
+---------+-----------+--------+----------+
| Model 1 |    0.78   |  0.74  |   0.76   |
| Model 2 |    0.82   |  0.74  |   0.75   |
| Model 3 |    0.82   |  0.75  |   0.76   |
+---------+-----------+--------+----------+
```

Typesetting math: 0%