You can contact me at kpardhan@sfu.ca

My local machine is an 6 year old i7 4790k, 24GB ram, and new-architecture medium-range RTX 3060 TI 8GB. The CuDNN backend was used with the higher level Tensorflow (this might be a "default"). The included code runs reasonable fast, though stability is not guaranteed for lower spec machines. All non TensorFlow code can be run on Collaboratory, though the TensorFlow code might not run due to Colab's memory limitations. **My dataset is very ram intensive**. The script needs 15gb, on the smaller dataset. Attempting to save models or data to disk has caused many crashes, yet it is very fast to run on my machine

Preprocessing.ipynb is for turning the keggle dataset to mine preprocessed form.
Machine learning.ipynb is a linearly executable to show my work.
Eval.ipynb is a minimal code of the above without outputs.
eval_faster_at_cost_of_score.py is the above but a 50/50 train test split. Note is does output plots which clog/pause the main thread.
I run it like "`python eval_faster_at_cost_of_score.py >> output.txt`". `I can take a while, especially the tf-df part. But in case of crash you may lose all data.`

LSTM weights as over 400mb to 1gb. They are not provided, as a deep learning capable machine can get a nearly as good fit in less then

Training with batch_size=256 is only essential for 10 epochs, and is 15 to 25 secounds per epoch. That is faster than I can download from google drive. Let alone the marker's work have having to match weights to dataset (and chance of error)
history = model.fit(X_train, y_train, validation_split=0.05, epochs=10, batch_size=256, verbose=1, callbacks=[lr, cp])

This original data is from: https://www.kaggle.com/stackoverflow/stacksample
preprocessed data can be obtained from:
https://drive.google.com/drive/folders/1N85lMKVfxQSIn7hfRzdzuXNo-fSUHGQ5?usp=sharing

note the MLPClassifier at the very end might takes a very long time. Only say to save it is via pickle, which is a security issue to share. You may wish to skim. I left comments with outputs

on the full data set with **Tokenized**
all data is in my paper. I don't have high decimal numbers saved.

on the full data set with **TfidfVectorizer**
min_df=200 # 8908 dims

Clf: ExtraTreesClassifier
Accuracy score: 0.22334586682765653
Recall score: 0.27799837100386887
Precision score: 0.7935911481842874
F1 score: 0.3873825144566118

Clf: RandomForestClassifier
Accuracy score: 0.27327359674546653
Recall score: 0.35183539734547675
Precision score: 0.7965338057751677
F1 score: 0.4543767120735285
0.4543767120735285Train score: 0.9516728624535316
Test score: 0.27327359674546653
Time taken for RandomForestClassifier was 539.40

Clf: MLPClassifier
Accuracy score: 0.3619516178728798
Recall score: 0.5404240943336851
Precision score: 0.7520324734933328
F1 score: 0.6189265432694974
0.6189265432694974Train score: 0.4378400585176204
Test score: 0.3619516178728798
Time taken for MLPClassifier was 165.39

on the top 3 score data set with **tokenized**

Clf: ExtraTreesClassifier
Accuracy score: 6.245169751520308e-05
Recall score: 9.295726412094229e-05
Precision score: 0.15857315788576337
F1 score: 0.0001857227195545714
Train score: 1.0
Test score: 6.245169751520308e-05
Time taken for ExtraTreesClassifier was 31.04

Clf: RandomForestClassifier
Accuracy score: 0.0006010975885838297
Recall score: 0.0005675285598962793
Precision score: 0.23681389711605078
F1 score: 0.00112958540555 4267
Train score: 0.8762490632025981
Test score: 0.0006010975885838297
Time taken for RandomForestClassifier was 41.99

Clf: MLPClassifier
Accuracy score: 0.0
Recall score: 0.0008023679639912914
Precision score: 0.04850754334439879
F1 score: 0.0015413825206534702
0.0015413825206534702
Train score: 0.0
Test score: 0.0
Time taken for MLPClassifier was 54.98

on the top 3 score data set with **TfidfVectorizer**
Clf: ExtraTreesClassifier
Accuracy score: 0.2486494925839188
Recall score: 0.29594565578090537
Precision score: 0.7973873461949164
F1 score: 0.40604024121287086
Train score: 1.0
Test score: 0.2486494925839188
Time taken for ExtraTreesClassifier was 157.88

Clf: RandomForestClassifier
Accuracy score: 0.30379391100702574
Recall score: 0.37855511526674346
Precision score: 0.7756578961481374
F1 score: 0.47787457166061137
Train score: 0.9530906025074553
Test score: 0.30379391100702574
Time taken for RandomForestClassifier was 177.91

Clf: MLPClassifier
Accuracy score: 0.3590007806401249
Recall score: 0.5912861855589607
Precision score: 0.6850863021757897
F1 score: 0.6321836389630043
Train score: 0.899163140720386
Test score: 0.3590007806401249
Time taken for MLPClassifier was 1085.79

The below are at over 40 epochs and after training embeddings, decreasing both batch size and LR. My eval code is focused on **speed**, you should get nearly as good performance.

Regarding the LSTM on the **all scores** dataset "questions_preprocessed.csv"

2963/2963 [==============================] - 85s 29ms/step - loss: 0.0199 - acc: 0.6533 - val_loss: 0.0242 - val_acc: 0.6135 - lr: 5.0000e-04

Accuracy score: 0.6504909819639279
Recall score: 0.7615028687766056
Precision score: 0.7574494222705919
F1 score: 0.7544918807953225

Regarding the LSTM on the **min 3 scores** dataset "questions_preprocessed_min3.csv"

1902/1902 [==============================] - 47s 25ms/step - loss: 0.0195 - acc: 0.6716 - val_loss: 0.0313 - val_acc: 0.5739 - lr: 1.5000e-04

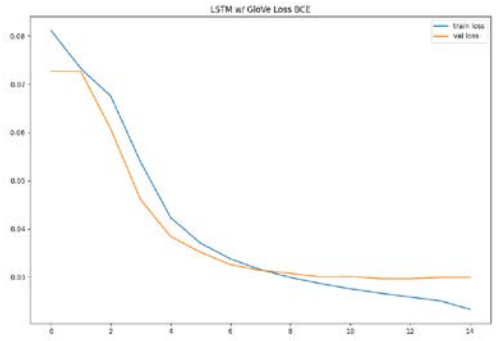
Accuracy score: 0.6123965651834504
Recall score: 0.7221690839173125
Precision score: 0.7139613396235852
F1 score: 0.714711189087093

Script runs fast on my machine.

Script output `eval_faster_at_cost_of_score.py`

```
alik604@BlueEyesPC MINGW64 /d/SFU Files/SFU Fall 2021/CMPT
413/Assigment_repoCSIL/nlpclass-1217-g-EZNLP/project (master)
$ python eval_faster_at_cost_of_score.py
100%|##########| 160123/160123 [00:35<00:00, 4524.01it/s]
100%|##########| 160123/160123 [00:00<00:00, 198172.89it/s]
2021-12-08 15:52:20.880437: I tensorflow/core/platform/cpu_feature_guard.cc:151] This
TensorFlow binary is optimized with oneAPI Deep Neural Network Library (oneDNN) to use
the following CPU instructions in performance-critical operations:  AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler
flags.
2021-12-08 15:52:23.134797: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1525]
Created device /job:localhost/replica:0/task:0/device:GPU:0 with 5497 MB memory:  ->
device: 0, name: NVIDIA GeForce RTX 3060 Ti, pci bus id: 0000:01:00.0, compute
capability: 8.6
2021-12-08 15:52:33.822561: I tensorflow/stream_executor/cuda/cuda_dnn.cc:366] Loaded
cuDNN version 8200
2021-12-08 15:52:39.037972: I tensorflow/stream_executor/cuda/cuda_blas.cc:1774]
TensorFloat-32 will be used for the matrix multiplication. This will only be logged
once.
length is 160123
Sample question: ASP NET Site Maps Has anyone got experience creating SQL based ASP
NET site map providers ve got the default XML file web sitemap working properly with
my Menu and SiteMapPath controls but ll need way for the users of my site to create
and modify pages dynamically need to tie page viewing permissions into the standard
ASP NET membership system as well

Number of tags: 2
Clf:  ExtraTreesClassifier
Accuracy score: 0.0001249024199843872
Recall score: 0.0001756817427628882
Precision score: 0.15602100372835698
F1 score: 0.0003507402672709734 7
Train score: 1.0
Test score:  0.0001249024199843872
Time taken for ExtraTreesClassifier was 50.70


Clf:  RandomForestClassifier
Accuracy score: 0.0019047619047619048
Recall score: 0.0016982568467079193
Precision score: 0.229359021908164
F1 score: 0.0033224858662357637
Train score: 0.8793892176302518
Test score:  0.0019047619047619048
Time taken for RandomForestClassifier was 85.48
```

```
Clf:  MLPClassifier
Accuracy score: 0.0
Recall score: 7.808077456128365e-05
Precision score: 0.05107458665989967
F1 score: 0.0001556840539053019
Train score: 0.0
Test score:  0.0
Time taken for MLPClassifier was 102.92
Accuracy score: 0.03731459797033568
Recall score: 0.12211833141384762
Precision score: 0.035572550178473994
F1 score: 0.04254961554891439

y_pred: c#
y_test: python


y_pred: c#
y_test: ios


y_pred: c#, java
y_test: oop, php


y_pred: c#
y_test: c#


y_pred: c#, java
y_test: facebook, ios


portion of words in embedding: 0.2708
Epoch 1/10
…

..

Accuracy score: 0.5880405932864949
Recall score: 0.7043081067364189
Precision score: 0.6854190453408421
F1 score: 0.6837298908463676
y_pred: android
y_test: android


y_pred: css, html
y_test: css, css3


y_pred: c, objective-c
y_test: c++, ios


Epoch 1/5
…

..
```



y_pred: mysql, postgresql, python, ruby, ruby-on-rails
y_test: database, mysql, postgresql

y_pred: android, c#, google-chrome, image, php
y_test: google-chrome

y_pred: .net, c#, generics, list, performance
y_test: .net, c#, generics

y_pred: css, html, javascript, jquery, twitter-bootstrap
y_test: html5, twitter-bootstrap

y_pred: c, c#, c++, delphi, java
y_test: c++

LSTM model scores
Accuracy score: 0.5958782201405152
Recall score: 0.7100275234730329
Precision score: 0.697738885183932
F1 score: 0.6931648053300155
Expirement 2: Encoding with TfidfVectorizer

The number of features is 3380
The number of samples is 160123
The number of features is 3380
Clf: ExtraTreesClassifier
Accuracy score: 0.2553942232630757
Recall score: 0.30502254582365457
Precision score: 0.7926059155735715
F1 score: 0.41497574107200613
Train score: 1.0
Test score: 0.2553942232630757
Time taken for ExtraTreesClassifier was 158.97

Clf: RandomForestClassifier
Accuracy score: 0.3110070257611241
Recall score: 0.38956450447988444
Precision score: 0.7770160992569072
F1 score: 0.48845383878318743
Train score: 0.953324798201377
Test score: 0.3110070257611241
Time taken for RandomForestClassifier was 182.45

Caution MLPClassifier can take long to train. I suggest max_iter=30 for performance, less for speed

```
Clf:  MLPClassifier
Accuracy score: 0.33224043715846996
Recall score: 0.5885923988365964
Precision score: 0.6553028068067148
F1 score: 0.618944669644878
Train score: 0.9952380208902559
Test score:  0.33224043715846996
Time taken for MLPClassifier was 1796.14
y_pred: javascript, jquery
y_test: jquery


y_pred: arrays, perl, ruby
y_test: arrays, ruby


y_pred: multithreading
y_test: python


y_pred: jquery, php
y_test: javascript, jquery, php


y_pred:
y_test: xml


Accuracy score: 0.4064949258391881
Recall score: 0.5257959358956841
Precision score: 0.688115162028338
F1 score: 0.5838141455326533
Train set
Accuracy score: 0.9952380208902559
Recall score: 0.9984299646864331
Precision score: 0.9985589895611524
F1 score: 0.9984937162356999
Test set
Accuracy score: 0.33224043715846996
Recall score: 0.5885923988365964
Precision score: 0.6553028068067148
F1 score: 0.618944669644878
```