Programming: Language Identification

1. Warm-up
   Size of vocabulary: 509
   Percentage of out of vocabulary tokens: 0.0469%
   Entropy for validation data is: 5.03
   Perplexity for validation data is: 32.65

```
Python 3.6.4rc1 (v3.6.4rc1:3398dcb, Dec  5 2017, 20:41:32) [MSC v.1900 64 bit (A
MD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:\Users\zhaoq\Desktop\EE511\HW#4\QIHAN_ZHAO_HW#4\Unigram\unigram.py
The size of vocabulary is 509
The percentage of out_of_dictionary is 0.046940268995445074%
The entropy for iid is 5.0289902738668975
The perplexity for iid is 32.64952898378137
>>>
```

2. Recurrent Neural Network
   ReadMe document explains how to run the code inside folder named as model1
   The model is character-based LSTM model, which embeds input characters and outputs predictions for next character.
   After training the model and validating on the validation data,
   the entropy for validation data is: 3.11
   the perplexity for validation data is: 8.65

```
train loss is: 3.0283995755016804
train perplexity is: 8.159040916845173
train loss is: 2.984604377299547
train perplexity is: 7.915082442112246
train loss is: 2.942539196461439
train perplexity is: 7.68763156947298
train loss is: 2.9043988455086946
train perplexity is: 7.487057558310888
train loss is: 2.869869861751795
train perplexity is: 7.309992174543409
train loss is: 2.8343938309699297
train perplexity is: 7.132430732547568
train loss is: 2.806098122149706
train perplexity is: 6.993904624181799
train loss is: 2.771389653906226
train perplexity is: 6.827652599555506
train loss is: 2.7452434562146664
train perplexity is: 6.705028462062569
valid loss is: 3.112209426239133
valid perplexity is: 8.64705835993656
```

3. Language identification
   ReadMe document explains how to run the code inside folder named as model2
   The model is also character-based LSTM model, but it embeds both input characters associated with the language id. It takes time to think about how to feed both a character and a language embedding and what is the output from RNN hidden layer at this time. I find the main idea is that based on given labels, the probability on predicting next characters varies. In the other word, a correctly assigned label will maximize the accuracy to predict next character and minimize the cross entropy. Based on designed model and feed both character associated with language

embedding, we assign the text with the label minimizing the cross-entropy. However, the trained model does not work very well for the validation set.
The test.ids.csv is in the folder named as model2, which produced by test.py.