ITCS 5111
Introduction to Natural Language Processing

Rahul Rachapalli
800968032

Problem 1 (Language Model Creation) (80 points).

In this exercise, you will train probabilistic language models to distinguish between words in different languages. Rather than looking up whole words in a dictionary, you will build models of character sequences so you can make a guess about the language of unseen words. You will need to use NLTK and the Universal Declaration of Human Rights corpus.

We will compare across different languages from the Universal Declaration of Human Rights documents. Use the following code to load the corpus and create sets of four languages.

```
import nltk
from nltk.corpus import udhr

english = udhr.raw('English-Latin1')
french = udhr.raw('French_Francais-Latin1')
italian = udhr.raw('Italian_Italiano-Latin1')
spanish = udhr.raw('Spanish_Espanol-Latin1')
```

If you do not have the UDHR dataset already installed with your version of NLTK, use nltk.download() to download the corpus.

Create training, development and test samples for English, French, Italian, and Spanish from these sets.

```
english_train, english_dev = english[0:1000], english[1000:1100]
french_train, french_dev = french[0:1000], french[1000:1100]
italian_train, italian_dev = italian[0:1000], italian[1000:1100]
spanish_train, spanish_dev = spanish[0:1000], spanish[1000:1100]
english_test = udhr.words('English-Latin1')[0:1000]
french_test = udhr.words('French_Francais-Latin1')[0:1000]
italian_test = udhr.words('Italian_Italiano-Latin1')[0:1000]
spanish_test = udhr.words('Spanish_Espanol-Latin1')[0:1000]
```

Build unigram, bigram, and trigram character models for all four languages. You may find it convenient to use the NLTK classes `FreqDist` and `ConditionalFreqDist`, described in chapter 2 of the NLTK book (http://www.nltk.org/book/).

For each word in the English test sets, calculate the probability assigned to that string by English vs. French unigram models, English vs. French bigram models, and English vs. French trigram models. Use the test set to report accuracy of your models. You should report the accuracies of the uni-, bi-, and tri-gram models.

Problem 2 (Language Model Comparison) (20 points).**

Perform the same experiment as above for Spanish vs. Italian. Which language pair is harder to distinguish?

Tabulate the final accuracies and their comparisons between the models.

The results for the language prediction of English Vs French are:

```
#################################################
#                    UniGrams                   #
#################################################
TestData\Model      English         French
English Test        78.2%           32.0%
French Test         26.6%           86.2%

#################################################
#                    BiGrams                    #
#################################################
TestData\Model      English         French
English Test        86.4%           23.8%
French Test         28.2%           84.6%

#################################################
#                    TriGrams                   #
#################################################
TestData\Model      English         French
English Test        86.6%           23.6%
French Test         29.1%           84.1%
```

The prediction results for Spanish Vs Italian are as follows

```
#################################################
#                   UniGrams                    #
#################################################
TestData\Model      Spanish         Iatlian
Spanish Test        68.5%           40.8%
Italian Test        39.7%           71.2%

#################################################
#                   BiGrams                     #
#################################################
TestData\Model      Spanish         Iatlian
Spanish Test        81.9%           27.4%
Italian Test        74.7%           74.7%

#################################################
#                   TriGrams                    #
#################################################
TestData\Model      Spanish         Italian
Spanish Test        60.3%           50.4%
Italian Test        74.9%           48.1%
```

The tables in the above results area analysed as follows

Test Data \ Model	English	French
English Test	Accuracy of model predicting ENGLISH given English and French Models for english test data.	Accuracy of model predicting FRENCH given English and French Models for english test data.
French Data	Accuracy of model predicting ENGLISH given English and French Models for french test data.	Accuracy of model predicting FRENCH given English and French Models for french test data.

Project Description:

1. The given training data is cleaned to remove punctuation and new line characters.
2. Using the obtained training data, a Unigram, Bigram and Trigram models are built.

Which language pair is harder to distinguish?

Spanish and Italian models are harder to distinguish. From the accuracy result, we see that in a trigram model, more than 28% of the spanish words are predicted to be Italian by the Italian trigram model and vice versa. The same in a unigram model leads up to 40%. Hence Italian and Spanish languages are hard to differentiate than English and French language pairs.

How did you calculate the probabilities for your models?

Unigram Model:

- The frequencies for all the characters are calculated.
- Prepare a dictionary based on the above output.
- For each letter in the dictionary, divide the frequencies by the total number of letter present in the training data.
- The resultant dictionary is a Uni-gram model.

Prediction using Unigram model:

- For each word in the test data, calculate the unigram probabilities using the chain rule.
- For example, probability for the word “hello” is

$$P(\text{“}hello\text{”}) = P(\text{‘}h\text{’}) * P(\text{‘}e\text{’}) * P(\text{‘}l\text{’}) * P(\text{‘}l\text{’}) * P(\text{‘}o\text{’})$$

- If the character is not available in the unigram, the smoothing value is returned.
- The same step is repeated for the 2 unigram models passed to the function.
- If the probability value of the 1st model is greater than the 2nd, the test word belongs to the first model.
- Accuracy is the total number of words that the model predicted accurately divided by the total number of words in the test set.Prepare a dictionary based on the above output.

Bigram Model :

- The training data is split into tuples with 2 adjacent values in the data.
- The frequency of occurrence of the joint pair of characters is determined and is updated in the dictionary.
- The resultant dictionary is a Bi-gram model.

Prediction using Bigram model:

- For each word in the test data, calculate the bigram probabilities using the chain rule.
- For example, probability for the word “hello” is

$$P(\text{“}hello\text{”}) = P(\text{‘}h\text{’}) * P(e/h) * P(l/e) * P(l/l) * P(o/l) * P(o)$$

- If the tuple does not appear in the Bigram dictionary, I have used the *Fall Back strategy* to determine the probability i.e., the unigram probabilities of

the two characters are returned. If the unigram probabilities are not available, smoothing value is returned.

- ➢ The same step is repeated for the 2 bigram models passed to the function.
- ➢ If the probability value of the 1st model is greater than the 2nd, the test word belongs to the first model.
- ➢ Accuracy is the total number of words that the model predicted accurately divided by the total number of words in the test set.Prepare a dictionary based on the above output.

Trigram Model :

- ➢ The training data is split into tuples with 3 adjacent values in the data.
- ➢ The frequency of occurrence of the joint pair of characters is determined and is updated in the dictionary.
- ➢ The resultant dictionary is a Tri-gram model.

Prediction using Bigram model:

- ➢ For each word in the test data, calculate the bigram probabilities using the chain rule.
- ➢ For example, probability for the word “hello” is

$$P(\text{“}hello\text{”}) = P(\text{‘}h\text{’}) * P(e/h) * P(l/eh) * P(l/le) * P(o/ll) * P(o/l) * P(o)$$

- ➢ If the tuple does not appear in the Triigram dictionary, I have used the *Fall Back strategy* to determine the probability i.e., the bigram probabilities of the combinations of the three characters are returned. If the above step fails, unigram probabilities of the three characters are returned. If the unigram probabilities are not available, smoothing value is returned.
- ➢ The same step is repeated for the 2 trigram models passed to the function.
- ➢ If the probability value of the 1st model is greater than the 2nd, the test word belongs to the first model.
- ➢ Accuracy is the total number of words that the model predicted accurately divided by the total number of words in the test set.Prepare a dictionary based on the above output.

Challenges Faced:

- ➔ Maintaining a multi level dictionary for Bigram and Trigram models.
- ➔ Implementing Fall back strategies
- ➔ Finding the initial probabilities in Bigrams and Trigrams
- ➔ Tried smoothing but, did not work out much.
- ➔ Coming up with a strategy to predict the probability of a tuple in a bigram and a trigram when it is not available in bigram and trigram respectively.