**Assignment2**: Language Models Applied to the Task of Language Identification

Issued: 02/11/2020

Due: 02/25/2020

Total points: 100

The goal of this assignment is to give you hands on experience with n-grams. Thus, you will implement and experiment with some form of statistical language modeling, and write a short report about your experiences and findings. As you know, n-gram language models could generate text or could be used to assess which sentence out of several (for example, generated by OCR, machine translation, etc.) is mostly likely. The parameters of this model will also be estimated from data.

Your specific task for this assignment is, however, Language Identification, which is the problem of taking as input a text in an unknown language and determine what language it is written in. N-gram models are very effective solutions for this problem as well.

For training, use the English, French, and Italian texts made available (see the Assignment2 folder). For test, use the file LangId.test provided in the Assignment2 folder as well. For each of the following questions, the output of your program has to contain a list of

[line\_id] [language] pairs, starting with line 1. For instance,

1 English

2 Italian

...

**Note:** The training files and the test file have been pre-processed for ease of manipulation (i.e., there is one sentence per line and each sentence is tokenized). Also, please note that these files use a Western encoding (python 3 uses Unicode (utf-8)).

1) **Question#1** [40 points].

Implement a letter bigram model, which learns letter bigram probabilities from the training data. Thus, a separate bigram model has to be learned for each language. Then apply the models to determine the most likely language for each sentence in the test file (that is, determine the probability associated with each sentence in the test file, using each of the three language models).

Design decisions:

1. What do you consider a token for this task and why (i.e., do you include punctuation marks as well)? What kind of preprocessing steps, if any, do you need to apply before you feed the data into your language model?
2. What technique do you decide to use for out of vocabulary (OOV) words and why?
3. Can the letter bigram model be implemented without any kind of smoothing? If not, use add-one smoothing. Is this kind of smoothing appropriate or you need better algorithms? Why (not)?

Compare your output file with the solution file (LangId.sol). How many times was your program correct?

Save the program as letterLangId.py and save the output as letterLangId.out

2) **Question#2** [30 points].

Implement a word bigram model, which learns word bigram probabilities from the training data. Again, a separate model will be learned for each language.

Design decisions:

1. What do you consider as a word for this task and why (i.e., only alpha-numeric characters or do you want to use punctuation marks as well as valid word tokens)? What kind of preprocessing steps, if any, do you need to apply before you feed the data into your language model?
2. What technique do you decide to use for out of vocabulary (OOV) words and why?
3. Can the word bigram model be implemented without any kind of smoothing? If not, try add-one smoothing. Is this kind of smoothing appropriate or you need better algorithms? Why (not)?

Apply the models to determine the language for each sentence in the test file.

Compare your output file with the solution file provided in the Assignment2 folder (LangId.sol). How many times was your program correct?

Save the program as wordLangId.py and save the output as wordLangId.out.

3) **Question#3** [30 points].

Same as Question#2, point c), but replace the add-one smoothing with Good-Turing smoothing. What do you do when the number of words seen once are unreliable? What strategy do you use to smooth unseen words?

Save the program as wordLangId2.py and save the output as wordLangId2.out.

Which of the language models at Question#1, Question#2, and Question#3 is the best? Comment on advantages and disadvantages of these language models on the task (be as detailed as possible based on your observations).

**Deliverables:**

* provide a README file including a detailed note (i.e., one paragraph) about the functionality of each of the programs, and complete instructions on how to run them; make sure you include your name in each program and in the README file; make sure all your programs run correctly.
* Provide an answer.pdf file where you should include answers to all the questions above with special focus on design decisions.

**NOTE:**

- you are NOT allowed to use any *language model toolkits or libraries* (including the ones provided with NLTK, spaCy, CoreNLP, Gensim, Pattern, PyNLPI, TextBlob, etc.) for this assignment: use pure python functions. You have to implement the models yourself. It is ok to discuss the design decisions among yourselves, but the design justifications and the code have to be yours, alone (i.e., this is an individual assignment)!