**Assignment2**: Language models applied to the task of Language Identification

Issued: 02/06/2018

Due: 02/20/2018

Total points: 80

The goal of this assignment is to get 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 predict the next word given the previous n-1 words. This model could generate text (as shown in the examples in class), 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 have been 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

...

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).

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

Can the letter bigram model be implemented without any kind of smoothing? Yes/no? What do you decide to do and why did you do it that way?

Save the program as letterLangId.py (if you don’t want to use python, use any programming language you like (we do prefer Python!!!))

Save the output as letterLangId.out

2) Question#2 [40 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. Use add-one smoothing to avoid zero-counts in the data. 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 (again, use any programming language of your choice)

Save the output as wordLangId.out

3) Analysis:

Which language model at Question#1, Question#2 is the best? Comment on advantages and disadvantages of these language models on the task.

Don’t do shallow work: be as detailed as possible based on your observations.

EXTRA-CREDIT QUESTION:

3) Question#3 [20 points].

Same as Question#2, but replace the add-one smoothing with any other better smoothing technique mentioned in class/textbook (e.g., Good-Turing, back-off/interpolation).

Save the program as wordLangId2.py

Save the output as wordLangId2.out

Which language model at Question#1, Question#2, and Question#3 is the best? Comment on advantages and disadvantages of these language models on the task.

Don’t do shallow work: be as detailed as possible based on your observations.

**Submission instructions**:

- write a README file including a detailed note about the functionality of each of the programs, and complete instructions on how to run them; the README file should also include answers to all the questions above.

- make sure you include your name in each program and in the README file.

- make sure all your programs run correctly on the any on-campus machine(s) of your choice (you will also be given partial credit if they don’t, but try your best!!!)

More in class about some issues:

* Programming language to use: python vs. C/C++ and others
* No NLTK bigrams allowed!!!
* Smoothing and numerical underflow (take care of your probabilities!!!)
* Running time