Machine Learning for Natural Language Processing
Execrice 1

The goals of this exercise are to analyze some statistical properties of natural language, build language models and use them for classification.

Data

We will use eight books written by Edgar Rice Burroughs: "The Chessmen of Mars", "The Gods of Mars", "Tarzan the Terrible", "Tarzan of the Apes", "The Return of Tarzan", "The Beasts of Tarzan", "The Son of Tarzan" and "Tarzan and the Jewels of Opar".

Download the data from This link. The subdirectory `books` contains one file per chapter, of the format `BookId.ChapterId.txt`. The subdirectory `train` contains few files per book, one per chapter. We use file naming of the form `BookID.ChapterID.txt`. Each paragraph of the original text is presented in a single line, all punctuation marks are removed and upper case letters are transformed into lower case. The subdirectory `evaluation` contains the remaining chapters per book, the subdirectory `evaluation_chapters` contains the same information, but each chapter is put in a single (long) line.

Code

The file `make_data_class.py` was used to generate the data (no need to run it, the data is already generated). The file `process_class.py` solves some of the items below. You are free build on it and/or modify it to solve the remaining items.

Task I : Statistics

We first explore some statistical properties of the data. Perform the next tasks per book.

1. Write a program that calculates the number of times each token was observed.
2. Use the Good-Turing method to estimate how many tokens were *not* observed for each book. Compare with the actual numbers of new tokens in the `evaluation` directory.
3. Order all the tokens according to their frequency, most frequent token has a rank of 1, the second frequent token has the rank of 2 and so on. Normalize the counts to have a proper distribution over ranks. Find the best fit of this empirical distribution to the following parametric families:
   - Geometric: $p(R) = (1-p)^{R-1}p$

- Poisson :
$$p(R) = \frac{\lambda^R e^{-\lambda}}{R!}$$
- Zipf
$$p(R) = \frac{1/R^a}{\sum_{r=1}^{\rho}(1/r)^a}$$
(try also $\rho = \infty$ which is feasible only a subset of the values of a).

Plot the distribution for the best choice of parameters for each of the three families as well as the empirical distribution. Conclude what is the best model for the data.

4. Given the best model estimate the probability of observing a *new* token in the next draw.

Task II : Language Modeling

1. Build eight simple Unigram models, one per book, of the form
$$P_1(w) = \frac{C(w)}{\sum_u C(u)}.$$

2. Compute the perplexity for each *line* in each file in the `evaluation` directory. Compare the average perplexity per book per model (eg in a 8×8 table) and compare how well each model is good for the book it models as well as other books.

3. Repeat the last task per book, use the file in `evaluation_chapters`. Note: nothing to average now, compute one number per file per model.

4. The last model is not applicable for some of the data (why?). Improve it using three methods, the first two we learned in class:

   - Laplace smoothing:
   $$P_2(w) = \frac{C(w) + a}{\sum_u C(u) + a * N}$$
   where a is the smoothing parameter and N is the total number of tokens. Play with different values of a and N (which should be larger than the number of tokens observed.)
   - Simple interpolation:
   $$P_3(w) = (1 - \delta)P_1(w) + \delta\frac{1}{N}$$
   where $\delta \in (0, 1)$ is the smoothing parameter and N is the total number of tokens. Play with different values of δ and N (which should be larger than the number of tokens observed.)
   - Additional method of your choice, either learned in class, or other.

5. Repeat the last items using each of the new models. What method performs the best? Are the methods sensitive to the choice of parameters? Compare the methods also using the entropy of the models.

6. Analyze the perplexity vs the length of the sentence (e.g. using histogram, or cumulative analysis). What are easier to generate: longer or shorter sentences?

Task III : Source Recognition

Evaluate the models from the last task on the a recognition task. Given a model and some text (eg line or entire chapter), compute the (log) probability of that text using each of the models and predict the book which its model obtained the highest probability.

1. Evaluate the models from the previous task on the recognition task on the sentence level and chapter level (I.e. use the two subdirectories `evaluation` and `evaluation_chapters`).
2. Compute the accuracy of each model as well as the confusion matrix.
3. Compare the evaluation of the language models using the perplexity and the recognition task. Is good perplexity indicates good classification?
4. Analyze the accuracy vs the length of the sentence (e.g. using histogram, or cumulative analysis). What are easier to predict: longer or shorter sentences?

Grading and Submission

Submit your results and discussion as well as code.

While you are building the language models and apply them to the various tasks above, hopefully better models will yield lower perplexity and better accuracy. Yet, the actual performance is not a major part of the grade, although good or better performance should indicate that things are correct, and bad performance may indicate that something is not working well.

Your grade will be based on your ability to make sense of the results you get. If one model is better, try to explain why? what types of mistakes it does improve. Apply observations you have from early parts to your work in later parts. If there are mistakes on some sentences please exemplify from the actual data.

Coding: It is recommended to code with languages that can manipulate text easily, such as python. You will be evaluated on natural language processing, not programming languages.

Extra Credit

Extra credit will be given for substantial additional work beyond the requirements above, especially in language modeling. E.g. using Bigrams.