CS190I Introduction to NLP Assignment 1: Unix Text Processing and Language Modeling

Due: 1/19, 10:59pm PT, submit to Gradescope

Late submission due: 1/23, 10:59pm PT

Instructor: William Wang
TAs: Yi-Lin Tuan, Wanrong Zhu

1 Policy on Collaboration among Students

We follow UCSB's academic integrity policy from UCSB Campus Regulations, Chapter VII: "Student Conduct and Discipline"):

"It is expected that students attending the University of California understand and subscribe to the ideal of academic integrity, and are willing to bear individual responsibility for their work. Any work (written or otherwise) submitted to fulfill an academic requirement must represent a student's original work. Any act of academic dishonesty, such as cheating or plagiarism, will subject a person to University disciplinary action. Using or attempting to use materials, information, study aids, or commercial "research" services not authorized by the instructor of the course constitutes cheating. Representing the words, ideas, or concepts of another person without appropriate attribution is plagiarism. Whenever another person's written work is utilized, whether it be a single phrase or longer, quotation marks must be used and sources cited. Paraphrasing another's work, i.e., borrowing the ideas or concepts and putting them into one's "own" words, must also be acknowledged. Although a person's state of mind and intention will be considered in determining the University response to an act of academic dishonesty, this in no way lessens the responsibility of the student."

More specifically, we follow Stefano Tessaro and William Cohen's policy in this class:

- You cannot copy the code or answers to homework questions or exams from your classmates or from other sources;
- You may discuss course materials and assignments with your classmate, but you cannot write anything down.
- You must write down the answers / code independently.
- The presence or absence of any form of help or collaboration, whether given or received, must be explicitly stated and disclosed in full by all involved, on the first page of their assignment.

Specifically, each assignment solution must start by answering the following questions:

1. Did you receive any help whatsoever from anyone in solving this assignment? Yes / No.
   If you answered 'yes', give full details: ____________________
   (e.g. "Jane explained to me what is asked in Question 3.4")
2. Did you give any help whatsoever to anyone in solving this assignment? Yes / No.
   If you answered 'yes', give full details: ____________________
   (e.g. "I pointed Joe to section 2.3 to help him with Question 2".)

Academic dishonesty will be reported to the highest line of command at UCSB. When you are not sure, ask the teaching staff before you do so. Students who engage in plagiarism activities will receive an F grade automatically.

2 Algorithms for Finding Top-100 Frequent Bigrams (27 points)

In the second class, we learned and practiced some simple Unix commands to find the top-100 frequent words from three BBC sports news articles. In addition to unigrams, bigram statistics are also essential for corpus analysis. Using the same text documents (e.g., 001.txt, 002.txt, 003.txt) from Piazza's resource page[1], please describe an algorithm for finding top-100 frequent bigrams using Unix commands only. **Describe each step of your algorithm in plain English, write out the exact commands, and show the first five lines of the output for each step.** You do not need to use pipe to complete this task in one line.

3 N-gram: the longer the better? (7 points)

In class, we discussed how to estimate unigram and bigram probabilities from text, and used those statistics to approximate the sentence probability. However, unigram and bigram methods are known to miss out long-range dependencies among words. **Question: can we use 10-gram or even longer n-gram models to model language? Why or why not?**

4 Language Modeling (LM) (66 points + 8 extra credits)

On the remote island of Banaanaa, the local aboriginal tribe uses a unique language that only includes three words: A, B, N. To save this endangered language, we, a team of computational linguists from UCSB, decided to build a language model to analyze the language. We sent our TAs to the island, and here's the training corpus they collected:
AAABANBABBBNNANBNN

4.1 Unigram Probabilities (16 points)

First, we consider training a simple unigram LM using maximum likelihood estimation (MLE). Report the probabilities for each question below (in fractions). In this particular problem, do not add start or end symbols.

4.2 Bigram Probabilities (24 points)

Train a bigram LM using using MLE. For this specific problem, we add an end symbol #. Do not add start tokens. Report the probabilities (in a fraction) for the questions below.

4.3 Testing and Perplexity (PPL) (26 points)

We decided to send our instructor to the remote island to collect the test data and examine our LM. Here's the test data he brought back:
ABANABB

1. Report the perplexity of the unigram LM. Note: do not add any tokens to the test data, just for the sake of consistency.
2. Report the perplexity of the bigram LM. Note: also add an end symbol # to the end to make it consistent.

4.4 Add-one Smoothing (8 extra credits)

We now perform add-one smoothing during training. Please report the new perplexities (from the previous question) using the smoothed versions of both unigram and bigram models.

[1] https://piazza.com/ucsb/winter2022/cs190i/resources

5 Linear Interpolation (8 extra credits)

Linear interpolation is a common method to combine statistical models in NLP. In the class, we talked about tuning the hyperparameter λs for n-gram language model interpolation using the holdout validation dataset. What will happen if we just tune those λs for an interpolated unigram+ bigram+trigram LM on the entire training set? What will the λs look like? Assume that you only have the access to a training set, can you design a better method for tuning λs, rather than just using the whole training set?

6 Out-of-vocabulary Words (OOVs) (4 extra credits)

A common problem for language modeling is that, after training and deployment, the system will encounter new, out-of-vocabulary words in the testing environment. For these OOVs, if we use standard training methods for bigram LM without smoothing, did we over-estimate or under-estimate the probabilities for OOV words during training? Can you design a training method to assign probabilities to sentences with a lot of OOV words during testing time? Be creative.

7 Submission Instructions

Put your solutions into one PDF file, and submit it to Gradescope at https://www.gradescope.com/courses/345550.