Course Policy: Read all the instructions below carefully before you start working on the assignment, and before you make a submission. The course assignment policy is available at http://nlp.cs.princeton.edu/cos484.

- This assignment contains 3 problems.
- We highly recommended that you typeset your submissions in $\LaTeX$. Use the template provided on the website for your answers. Include your name and NetIDs with your submission. If you wish to submit hand written answers, you can scan and upload the pdf.
- Assignments must be uploaded to Gradescope by 11:59pm on the due date mentioned above.
- Late submissions will incur a penalty of 10% for each day, up to a maximum of 4 days beyond which submissions will not be accepted.
- One of the goals of this assignment is for you to assess whether you have adequate preparation for the course. Its fine to not be familiar with every concept here. However, if you find yourself struggling with much of this assignment (especially Problems 1 and 2), you should ask the course staff whether this course is appropriate for you given your background.

Problem 1: Preliminaries (5 x 1 + 1 + 1 + 3 = 10 points)

(a) (Linear Algebra) Provide answers to the following operations or write "invalid" if not possible:

(i) $\begin{bmatrix} 3 & 5 \\ 4 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 7 \end{bmatrix}$

(ii) $\begin{bmatrix} 3 & 5 \\ 4 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 7 \end{bmatrix}$

(iii) $\begin{bmatrix} 3 & 5 & 4 \\ 4 & 1 & 1 \end{bmatrix} \begin{bmatrix} 4 & 3 & 2 \end{bmatrix}^T$

(iv) $\begin{bmatrix} 3 & 5 & 4 \end{bmatrix}^T \begin{bmatrix} 4 & 3 \end{bmatrix}^T$

(v) $\begin{bmatrix} 3 & 5 & 4 \end{bmatrix}^T \begin{bmatrix} 4 & 3 \end{bmatrix}$

(b) (Probability) The entropy of a discrete random variable $X$ is defined as (use base $e$ for all log operations unless specified otherwise):

$$H(X) = - \sum_{x \in X} P(x) \log P(x)$$

(i) Compute the entropy of the distribution $P(x) = \text{Multinomial}([0.2, 0.3, 0.5])$.

(ii) Compute the entropy of the uniform distribution $P(x) = \frac{1}{m} \forall x \in [1, m]$.

(iii) Consider the entropy of the joint distribution $P(X, Y)$:

$$H(X, Y) = - \sum_{x \in X} \sum_{y \in Y} P(x, y) \log P(x, y)$$

How does this entropy relate to $H(X)$ and $H(Y)$, (i.e. the entropies of the marginal distributions) when $X$ and $Y$ are independent?
Problem 2: Language models and perplexity

Assume you are given the following corpus of text:
<s> I like fruits <e>
<s> You like bananas <e>
<s> I hate green bananas <e>
<s> You like green fruits <e>

where <s> and <e> are tokens representing the start and end of a sentence, respectively.

(a) Provide the equation for bigram probabilities and estimate all non-zero bigram probabilities for this corpus.

(b) Using the above estimate, provide the probabilities for the sentences <s> I hate green bananas <e> and <s> You hate green bananas <e>. Since the former is one of the four sentences in the corpus, would you expect its probability to be 0.25? Why or why not?

(c) For any generic corpus, can you estimate accurate trigram probabilities directly from bigram probabilities (without access to the true bigram/trigram counts)? Do you require any additional assumptions to be made on the data distribution? Justify your answer mathematically. Does our corpus above satisfy these assumptions?

(d) Provide the formula to calculate the perplexity of a corpus. Calculate the perplexity of this corpus using your probability estimates from 2(a):
<s> I like green fruits <e>
<s> You like fruits <e>

Problem 3: Smoothing in language models

This part will require programming in Python 3. You will explore building and testing a language model with smoothing. First, download the starter kit code provided. Once you unzip the archive, you will obtain two folders – data/ containing the Brown corpus and src/ containing basic skeleton code.

The code is split up into two files – data.py which contains functions for reading and processing the corpus, and lm.py for training and testing a language model. You can run the code as python src/lm.py [--args]. If you don’t have numpy/tqdm, you can install the required packages using pip install -r requirements.txt.

For all the coding parts below, you should not need to create any new files. Upload a single zip file containing both src/ and data/ folders to Gradescope under the corresponding programming assignment.

(a) Complete the functions nltkTokenize and countTopWords in data.py. Report the top 10 words ordered by their frequency in the training corpus, both using basicTokenize and nltkTokenize. What differences do you notice between the two?

(b) Using the nltkTokenize function you wrote, make a plot of the frequencies of each word in the training corpus, ordered by their rank, i.e. the most frequent word first, the second most word next, and so on on the x axis. What pattern do you observe in your plot? Do you see any deviations from Zipf’s law?

Use the basicTokenize function for the following questions.

(c) Run lm.py to train the model and report its perplexity on the train and validation sets. What do you notice?

(d) Implement Laplace (add-α) smoothing within the appropriate function provided (computeBigramAddAlpha in lm.py) and retrain the model. Plot the perplexity on train and validation sets as a function of alpha (with values 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10). What differences do you observe compared to the previous case? Include your plot in your submission.
(e) Fix $\alpha = 0.001$. Now vary the fraction of training data used (in increments of 10% from 10% to 100%) using the --train_fraction command line argument. Plot the train and test perplexities across this variation. Do the curves match your intuition?

(f) Now implement interpolation smoothing (function `computeBigramProbInterpolation` in `lm.py`):

$$P_{\text{int}}(w_t|w_{t-1}) = \beta P_2(w_t|w_{t-1}) + (1 - \beta)P_1(w_t)$$

where both $P_2$ and $P_1$ are Laplace-smoothed language models with $\alpha = 0.001$. Plot the train and validation perplexities as a function of $\beta$ (with values $0.1, 0.3, 0.5, 0.7, 0.9$) and include the plot in your submission. If you had to choose one model, which of these $\beta$ values would you choose and why?

**Important:** For all your programming exercises, do not make any modifications to the command line options and the training and testing loops. Make sure your code runs as expected before uploading. For example, the following types of commands should work:

```python
python src/lm.py
python src/lm.py --smoothing addAlpha --smoothing_param 0.001
python src/lm.py --smoothing interpolation --smoothing_param 0.01
```