WBAI059-05: **Natural Language Processing**

Assignment #1

Instructor: Tsegaye Misikir Tashu   65 points

Read all the instructions below carefully before you start working on the assignment and before you make a submission.

- This assignment contains 2 parts, a theoretical and a programming part.

- We *highly* recommended that you typeset your submissions in LaTeX for the theoretical part. Include your name and **student number** with your submission. If you wish to submit hand-written answers, you can scan and upload the pdf.

- Assignments must be uploaded to BrightSpace **before** the deadline. Assignments submitted after the deadline are not considered as a valid submission.

- Late submissions will incur a penalty of 10% for each day, up to a maximum of 2 days beyond which submissions will not be accepted.

- All programming problems will be completed in Google Colab using Python.

Assume you are given the following corpus of text:
```
<s> I like blueberries <e>
<s> You like all types of berries <e>
<s> I hate bitter fruits <e>
<s> You like all sweet fruits <e>
<s> I like chocolate covered raspberries <e>
```

where `<s>` and `<e>` are tokens representing the start and end of a sentence, respectively. In all the following questions, **ignore** probability of `<s>`.

(a) Provide the equation for bigram probabilities and estimate all **non-zero** bigram probabilities for this corpus (split into tokens by whitespace only, e.g `raspberries` and `berries` are different t okens). P resent the probabilities in a tabular format:

Bigram	Probability
$\mathbb{P}(\text{I} \mid \text{<s>})$	.
.	.

(b) Using the above estimate, provide the probabilities for the sentences `<s> You like all types of berries <e>` and `<s> You hate bitter fruits <e>`. Since the former is one of the five sentences in the corpus, would you expect its probability to be $\frac{1}{5}$? Why or why not?

(c) For any generic corpus, can you calculate accurate trigram probabilities for all word combinations $\mathbb{P}(w_3 \mid w_1, w_2)$ if you are given all bigram probabilities - $\mathbb{P}(w_3 \mid w_2)$ (without access to the true bigram/trigram counts)?

(d) Calculate the perplexity of the following corpus using your bigram model. Remember that `<s>` should not be included when calculating the length of the sentence.
```
<s> I like all sweet fruits <e>
<s> You like raspberries <e>
```

Problem 2: Smoothing in language models $\qquad$ $(5 + 5 + 5 + 10 = 25$ points)

This part will require programming in Python. You will explore building and testing a language model with smoothing.

For all the coding parts below, **you should not need to create any new files or notebooks**. The notebook has sections where you can fill in the code for all subproblems. Feel free to add and delete arguments in function signatures, but be careful that you might need to change them in function calls which are already present in the notebook.

(a) Complete the functions `nltkTokenize` and `countTopWords`. 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 words 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. Plot only the top 100 most common words to see the trend more clearly. What pattern do you observe in your plot regrading frequency and rank?

Use the `basicTokenize` function and bigram language model ($n = 2$) for the following questions.

(**c**) Train the model and report its perplexity on the train and validation sets. Is the train or val perplexity higher and why? What do you notice about the val perplexity and why is this the case?

(**d**) Implement Laplace (add-α) smoothing within the appropriate function provided (`computeBigramAddAlpha`) 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 happens to the validation and training perplexity as we increase alpha and why does this happen? What seems to be a good setting for alpha? Provide brief justification. Include your plot in your submission.

This problem will require programming in Python. The goal is to build a logistic regression model on the sentiment classification dataset.

The dataset you will be using is collected from movie reviews online. All the examples have been already tokenized and lowercased and each example is labeled as 1 (positive) or 0 (negative). The dataset has been split into a training, and a test set.

For all the coding parts below, **you should not need to create any new files or notebooks**. The notebook has sections where you can fill in the code for all subproblems. Feel free to add and delete arguments in function signatures, but be careful that you might need to change them in function calls which are already present in the notebook.

You will now build the key parts of a Logistic Regression model for sentiment analysis. You are required to implement the train lr function, LogisticRegressionClassifier class, and code for plotting accuracy when the regularization parameter is varied.

- train_lr: it takes a list of training examples (class SentimentExample) and a feature extractor (class FeatureExtractor) as input and is required to output an instance of LogisticRegressionClassifier.

- LogisticRegressionClassifier: you should at least implement __init__, train, and predict functions. Feel free to add more arguments to __init__ and train but don't change the arguments in predict — it should just take an example ex (class SentimentExample) and predict a label 1 or 0.

- You will probably want to take a look at the UnigramFeatureExtractor and BigramFeatureExtractor classes that we have implemented for you already!

(a) First, implement a logistic regression model without regularization. And then train your model with unigram and bigram features (code for creating features has already been written for you).

Report both training and development accuracy on the dataset. How do you compare the results of the unigram and bigram models? *Hint: You might find that batch optimization is too slow. Try to use stochastic gradient descent or (mini-batch) stochastic gradient descent!*

(b) Next, we would like to experiment with l_2 regularization $R(\theta) = \alpha\|\theta\|^2$. Plot the accuracy on train and development sets as a function of $\alpha = \{0, 10^{-2}, 10^{-1}, 1, 10\}$. You only need to experiment with unigram freatures for this part. Explain what you observe. Does this match what you would expect from regularization? *Hint: There are several ways to implement the logistic regression model with regularization, and we will be accepting any correct implementation — regardless of the final model performance.*