Assignment 1

COMP7607: Natural Language Processing - University of Hong Kong

Fall 2022

Zoie uses English in her everyday work. She finds it sometimes difficult to decide which preposition she should use in some cases. For example, should she say "schedule a meeting in 3pm" or "schedule a meeting at 3pm"? One day she visited HKU and came across a student who happens to study COMP3361 this semester.

"You must be an expert in NLP!", said Zoie.

"Well, I only got to know what is language model last week", said the student.

"That's more than enough", said Zoie with her eyes shining like stars.

"Could you please write a program for me that tells me what preposition I should use in a sentence?"

"Sure! My pleasure!", said the student.

And here comes assignment 1.

Reading Materials: We recommend that you read J&M Ch. 3 and M. Collins, Notes 1 before doing this assignment.

Resources: You can access all related resources at `https://github.com/ranpox/comp7607-fall2022/tree/main/assignments/A1/`.

Code: We provide a Jupyter Notebook template `https://github.com/ranpox/comp7607-fall2022/blob/main/assignments/A1/A1.ipynb`. You can use it on your local environment or Google Colab (recommended) `https://colab.research.google.com/github/ranpox/comp7607-fall2022/blob/main/assignments/A1/A1.ipynb` to finish it. Some useful packages you may need: `nltk`, `numpy`

Grade: We will evaluate your grade based on the code quality, output results/log, and discussion. We don't evaluate your submission by perplexity in Section 1, but better prediction accuracy in Section 2 increases your score. There are some optional problems in this assignment, and solving them increases your grade. Please don't waste your time **manually** searching for hyperparameters. You are more than welcome to introduce a new hyperparameter optimization method to find more reasonable values.

Submit: You should submit the **UniversityNumber.ipynb** file and final prediction file **UniversityNumber.test.out** of Section 2 to moodle. All code and discussion should be included in your submitted `.ipynb` file. Make sure your code does not use your local files so that the results are reproducible. Before submitting, please **run your notebook and keep all running logs** so that we can check the results.

1 n-gram Language Model

In this section, you will build your own language model. We provide a dataset with three files containing many whitespace-tokenized sentences at https://github.com/ranpox/comp7607-fall2022/tree/main/assignments/A1/data/lm:

- `train.txt`: data for training your language model.
- `dev.txt`: data for developing and optimizing your language model.
- `test.txt`: data for only evaluating your language model.

You will first build vocabulary with the training data, and then build your own unigram, bigram, and trigram language models with some smoothing methods.

1.1 Building Vocabulary

You will download and preprocess the tokenized training data to build the vocabulary. To handle out-of-vocabulary(OOV) words, you will convert tokens that occur less than three times in the training data into a special unknown token `<UNK>`. You should also add start-of-sentence tokens `<s>` and end-of-sentence `</s>` tokens.

Please show the vocabulary size and discuss the number of parameters of n-gram models.

1.2 n-gram Language Modeling

After preparing your vocabulary, you are expected to build bigram and unigram language models and report their perplexity on the training set, and dev set. Please discuss your experimental results. If you encounter any problems, please analyze them and explain why.

1.3 Smoothing

In this section, you will implement two smoothing methods to improve your language models.

1.3.1 Add-one (Laplace) smoothing

Please improve your bigram language model with add-one smoothing. Report its perplexity on the training set and dev set. Briefly discuss the differences in results between this and the bi-gram model you built in Section 1.2.

Optional: Add-k smoothing. One alternative to add-one smoothing is to move a bit less of the probability mass from the seen to the unseen events. Instead of adding 1 to each count, we add a fractional count k (.5? .05? .01?). This algorithm is therefore called add-k smoothing.

$$P^*_{\text{Add-k}}\left(w_n \mid w_{n-1}\right) = \frac{C\left(w_{n-1}w_n\right) + k}{C\left(w_{n-1}\right) + kV} \tag{1}$$

Please optimize the perplexity on the dev set by trying different k (no more than three times). Report its perplexity on the training set and dev set. Briefly discuss the differences in results between this and the bi-gram model with add-one smoothing.

1.3.2 Linear Interpolation

Please implement linear interpolation smoothing between unigram, bigram, and trigram models. Report its perplexity on the training set and dev set. Please optimize on a dev set by trying different hyperpara-

meter sets. Finally, report the perplexity on the test set with the best hyperparameter set you get. Briefly discuss the results.

Optional: Optimization. So far, we manually choose the hyperparameter that maximizes the perplexity of the dev set and evaluates it on the test set. There are various ways to find this optimal set of hyperparameters. Do you know any other learning algorithms? Please give an example.

2 Preposition Prediction

In this section, you will use your language model to answer what preposition to use in a sentence. We provide you with three files at https://github.com/ranpox/comp7607-fall2022/tree/main/assignments/A1/data/prep

- dev.in: whitespace-tokenized input sentences for developing your language model, where these 5 prepositions(at, in, of, for on) are replaced with a special token <PREP>.
- dev.out: The correct answers for the dev set.
- test.in: Input sentences for testing in the same format as dev.in, which will be released a week before the deadline.

Your task is to use your language model to predict for each special token ⟨PREP⟩ in the validation (and soon the test) set, which preposition is the proper one there. You should optimize the accuracy of your model on the dev set and predict the prepositions of sentences in the test set. Please strictly follow the format of dev.out for your output test.out as well.

Please explain your method, report the accuracy of your model on dev sets, and discuss the results. Based on your hyperparameters and results, explain the relationship between the test perplexity model and performance in applications.

Good luck, Zoie!