CS7.401: Introduction to NLP | Assignment 1

Instructor: Manish Shrivastava

Deadline: January 28, 2022 | 23:55

General Instructions

1. The assignment must be implemented in Python
2. No standard library for calculating n-grams, tokenization or LMs should be used.
3. Submitted assignment must be your original work. Please do not copy any part from any source including your friends, seniors, and/or the internet. If any such attempt is caught then serious actions including an F grade in the course is possible.
4. A single .zip file needs to be uploaded to the Moodle Course Portal.
5. Your grade will depend on the correctness of answers and output. In addition, due consideration will be given to the clarity and details of your answers and the legibility and structure of your code.
6. Please start early, since no extension to the announced deadline would be possible.

1 Tokenization

You have been given a twitter corpus for cleaning. Your task is to design a tokenizer using regex, which you will later use for smoothing as well.

1. Create a Tokenizer to handle following cases:
   (a) Word Tokenizer
   (b) Punctuation
   (c) URLs
   (d) Hashtags (#manchesterisred)
   (e) Mentions (@john)
2. For the following cases, replace the tokens with appropriate placeholders:
   (a) URLs: <URL>
   (b) Hashtags: <HASHTAG>
   (c) Mentions: <MENTION>

Original	**Cleaned**
#ieroween THE STORY OF IEROWEEN! THE VIDEO -»»»»»»»»»»»»»»»» http://bit.ly/2VFPAV «« JUST FOR FRANK !!! ÃƒÂ§	<HASHTAG> THE STORY OF IEROWEEN! THE VIDEO -> <URL> < JUST FOR FRANK ! ÃƒÂ§

Example output after placeholder substitution

Apart from these, you are also encouraged to try other tokenization and placeholder substitution schemes based on your observations from the corpora used for the smoothing task to achieve a better language model. You'll find percentages, age values, expressions indicating time, time periods occurring in the data. You're free to explore and add multiple such reasonable tokenization schemes in addition from the ones listed above. Specify any such schemes you use in the final README.

2 Smoothing

You have been given two corpus : EuroParl corpus, and Medical Abstracts corpus. Your task is to design Language Models for both of these corpora using smoothing. Ensure that you use the tokenizer created in task 1 for this task.

1. Create language models with the following parameters:
   (a) On EuroParl corpus:
      i. LM 1: tokenization + 4-gram LM + Kneyser-Ney smoothing
      ii. LM 2: tokenization + 4-gram LM + Witten-Bell smoothing
   (b) On Medical Abstracts corpus:
      i. LM 3: tokenization + 4-gram LM + Kneyser-Ney smoothing
      ii. LM 4: tokenization + 4-gram LM + Witten-Bell smoothing
2. For each of these corpora, create a test set by randomly selecting 1000 sentences. This set will not be used for training the LM.
   (a) Calculate perplexity score for each sentence of EuroParl corpus and Medical Abstracts corpus for each of the above models and also get average perplexity score/corpus/LM on the train corpus
   (b) Report the perplexity scores for all the sentences in the test set. Report the perplexity score on the test sentences as well, in the same manner above.
3. Compare and analyse the behaviour of the different LMs and put your analysis and visualisation in a report

3 Corpus

The following corpora have been given to you for training:

1. Tweets corpus (A set of 2000 tweets)
2. Europarl Corpus (A subset of 20000 lines from English side of the dataset for English-French translation)
3. Medical Abstracts Corpus (A subset of 20000 sentences from English side of the English-German translation task for medical abstracts in WMT 2014)

The first one is to be used for figuring out rules for the tokenization task. Use 2 & 3 for training your LMs. Please download the corpus files from this link.

4 Submission Format

Zip the following into one file and submit in the Moodle course portal. Filename should be <roll number>_<assignment1>.zip, eg: 2021xxxxxx_assignment1.zip:

1. Source Code:
   (a) language_model.py: Runs the language model given the following:

```
$ python3 language_model.py <n_value> <smoothing type> <path to corpus>
```

Smoothing type can be k for Kneyser-Ney or w for Witten-Bell. On running the file, the expected output is a prompt, which asks for a sentence and provides the probability of that sentence using the two smoothing mechanisms. Therefore, an example would be:

```
$ python3 language_model.py k ./corpus.txt
input sentence: I am a man.
0.89972021
```

2. A text file containing cleaned output (with placeholders) for all the tweets stored in <roll number>_tokenize.txt
3. Report containing the perplexity scores of all LMs and your analysis of the results in a PDF:
   (a) For each LM, submit a text file with perplexity scores of each sentences in the following format:

```
avg_perplexity
sentence_1 <tab> perplexity
sentence_2 <tab> perplexity
..
```

   (b) Naming must be: <roll number>_LM1_train-perplexity.txt, <roll number>_LM1_test-perplexity.txt
4. README on how to execute the files, how to get perplexity of sentences along with any other information.

5 Grading

Evaluation will be individual and will be based on your viva, report, submitted code review. In the slot you are expected to walk us through your code, explain your experiments, and report. You will be graded based on correctness of your code, accuracy of your results and quality of the code.

Marks Distribution (out of 100):

1. Tokenization: 20 Marks
2. Smoothing: 70 Marks
   (a) Kneyser-Ney: 30 Marks
   (b) Witten-Bell: 40 Marks
3. Quality and Efficiency: 10 Marks

6 Resources

1. Chapter 3 of the Speech and Language Processing book by Jurafsky & Martin
2. An Empirical Study of Smoothing Techniques for Language Modeling for Witten-Bell Smoothing
3. Official Python documentation and testing playground for regex.