CS 565: Assignment-2

Language Modeling

Due Date: 20^{th} **March,** 2019

- Your assignment and project group remain the same.
- Each group will submit a report and code. The report will include a brief write up of the analysis performed and its results along with the relevant plots. The code will include the complete code that was used to produce the results. In the report also mention individual contributions.
- If there is any query related to assignments, please post/message via Canvas only.

Dataset

Download a subset of English Wikipedia available at one of the following links: (0), (1), (2), (3). You need to complete the assignment only using one of the above text corpora. To determine the corpus to use, add the roll numbers of your group members and perform a modulus 4 operation on the sum. (Note: This dataset is same as the what was shared for Assignment 1.)

Prepare the training, development and test set as follows: After sentence segmentation and word tokenization, randomly shuffle sentences and split them into two parts of 90% (part-1) and 10% (part-2) of all sentences. Part-2 constitutes the independent test set and will remain fixed. Part-1 will be used for training and development purpose.

N-gram language model

1. Implement a tri-gram language model using Backoff and Interpolation smoothing methods.
2. Use Part-1 to split the data into two parts of 90% (training data) and 10% (development/validation) set. Do this five times to prepare five different pairs of training and development sets.
3. Report model's performances in terms of the perplexity and Likelihood on a held-out (development/validation) sets. Do you obtain the same set of parameters?
4. Report models performance on independent test set.
5. Summarize results in the report including discussions on which smoothing methods had a least variance and what happens if only Laplace smoothing is used.

Neural probabilistic language model

Implement a hierarchical neural probabilistic language model as discussed in the class with the following configurations:

1. Word embedding of dimension: 50

2. Context window size: 5; this implies you have to predict next word based on five previous context words.
3. Hidden Layer size: 100
4. For generating hierarchical output layer of words, obtain pre-trained word embedding and perform hierarchical k-means (with k=2) clustering [basically recursively split word embeddings into two clusters until all individual word become a cluster.]

Repeat parts 2 - 4 of the N-gram language model problem description and report your results.
Also compare the performance of the N-gram language models and neural model.