NLU : Assignement - II

anirbanb@iisc.ac.in

1 Task - I

1.1 Description

In this task, given dataset *D2:Gutenberg Corpus*, the task is to experiment with neural language models and compare it with the classical ngram based language models (built as part of first assignment). In this task, I have implemented **Token-level** LSTM-based Neural Language Model. I have tried *Adam Optimizer* as well as *Gradient Descent Optimizer* as optimization techniques.

For different settings like learning rate, multi-level neural architecture, varying epochs and so on, trained the models first and then calculated perplexity based on train data and validation data. The final model have been chosen based on the best perplexity value obtained for the various settings in train data. It turns out that the 2-level architecture along with *Dropout* gives best results i.e. lowest perplexity value among all the models.

Now once I finalized my model, I tuned the hyperparameters for the neural model. I tried with varoius settings of hyperparameters and finally took the one which gave best perplexity. Some of the hyperparameters are *learning rate* for the optimizer, *epochs*, *batch size* etc.

The division of train and test set size I have used is as follows: *Train set - 40%, Development Set - 10% and Test set - 10%*

1.2 Results

The following two graphs shows the comparison of perplexity measures and accuracy of next predicted word for train, validation and test dataset.

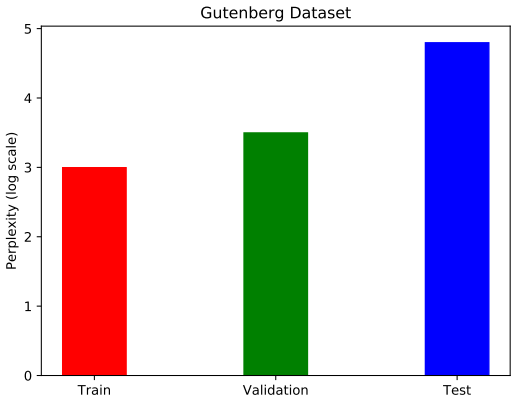

Figure 1: Perplexity measure on Gutenberg Dataset

As expected, train data gives minimum perplexity and best accuracy whereas test data gives maximum perplexity and least accuracy.

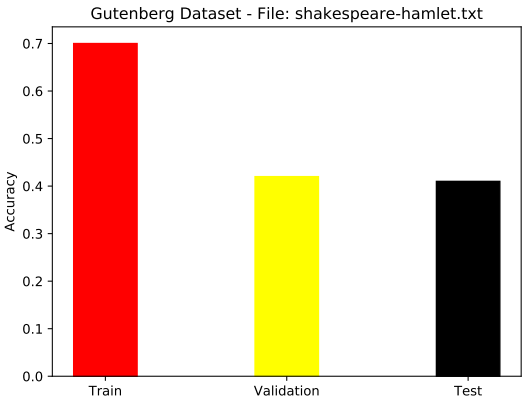

Figure 2: Accuracy measure on Gutenberg Dataset

1.3 Discussion

Perplexity and accuracy both depends on various hyperparameters used in the model. Among all the

hyperparameters I have tried to tune the model so that it gives best performance in development set, the following set of values seemed to give really good results.

1. **Batch Size** : 100
2. **Number of LSTM Units** : 650
3. **Number of steps** : 30
4. **Number of layers** : 2
5. **Keep Probability** : 0.35
6. **Learning rate** : 0.001
7. **Epoch** : 50

Based on the above set of hyperparameters, I have trained my model and then used it to calculate the perplexity for test data.

2 Task - II

2.1 Description

This task is similar to previous one. Given dataset *D2:Gutenberg Corpus*, the task is to design **Character-level** LSTM-based Neural Language Model and compare it with the classical ngram-based language model (built as part of first assignment) as well as Token-level neural model. I have used *Adam Optimizer* with static learning rate as optimization techniques.

For different settings like learning rate, multi-level architecture, varying epochs and so on, trained the models first and then calculated perplexity based on train data and validation data. The final model have been chosen based on the best perplexity value obtained for the various settings in train data. It turns out that the 2-level architecture along with *Dropout* gives best results i.e. lowest perplexity value among all the models.

Now once I finalized my model, I tuned the hyperparameters for the neural model. Parameter tuning is done based on validation data. I tried with various settings of hyperparameters and finally took the one which gave best perplexity. Some of the hyperparameters are *learning rate* for the optimizer, *epochs*, *batch size* etc.

The division of train and test set size I have used is as follows: *Train set - 40%, Development Set - 10% and Test set - 10%*

2.2 Results

The following two graphs shows the comparison of perplexity measures and accuracy of next predicted word for train, validation and test dataset.

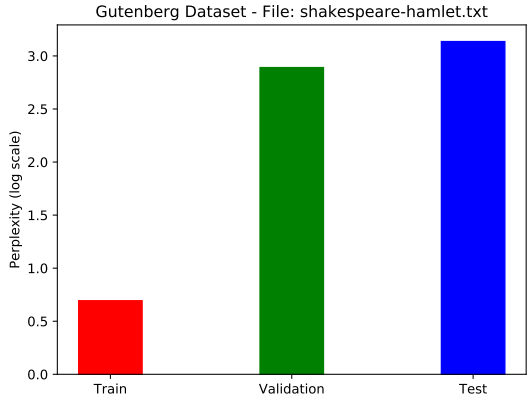

Figure 3: Perplexity measure on Gutenberg Dataset and File: shakespeare-hamlet.txt

As expected, train data gives minimum perplexity and best accuracy whereas test data gives maximum perplexity and least accuracy.

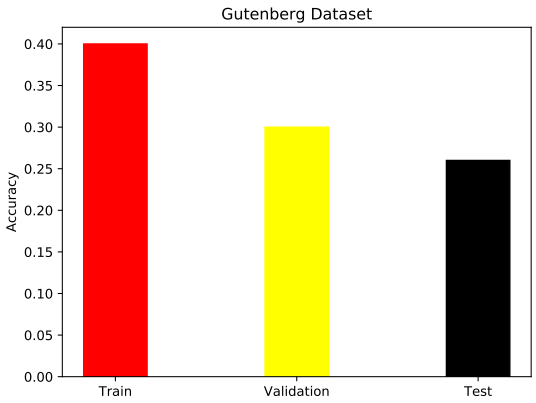

Figure 4: Accuracy measure on Gutenberg Dataset and File: shakespeare-hamlet.txt

2.3 Discussion

Perplexity and accuracy both depends on various hyperparameters used in the model. Among all the hyperparameters I have tried to tune the model so that it gives best performance in development set, the following set of values seemed to give really good results.

1. **Batch Size** : 20
2. **Number of LSTM Units** : 650
3. **Number of steps** : 30
4. **Number of layers** : 2
5. **Keep Probability** : 0.8
6. **Learning rate** : 0.005
7. **Epoch** : 20

Based on the above set of hyperparameters, I have trained my model and then used it to calculate the perplexity for test data.

3 Task - III

3.1 Description

Using the best model obtained from task-I and task-II, this task requires to generate a sentence of 10 tokens based on the trained model. Sentence generation task is done in such a way that it starts with the start sentence token. A next word is generated based on taking all the words with matching context and then selecting one word randomly. The next word is then generated in the same way, only the context now has been changed with the lastly added word comes into new context. Once 10 tokens are generated this process stops.

3.1.1 Final Model

Based on the perplexity values obtained for same set of training and test data, it is observed that **Word-level model** is performing best among the three (other two are Character-level model and classical ngram-based Language model). So, the word-level model has been chosen for the final task : generation of sentence.

Some example sentences generated by this model are - *" I have not been in a hurry , and I am sure , " And the very good - law is not a very agreeable young man* etc.