CS60075 : Assignment-1 Report

Debanjan Saha [19CS30014]

N-Gram Language Model:

I have trained unigram,bigram, trigram and 4-gram language models using 'train.txt' and noted down the perplexity of the test set for these models. Results are given as following:

>> Observed Perplexity:

Model_Type	Observed Test set Perplexity
Unigram	762.939
Bigram	85.795
Trigram	51.555
4-gram	40.549

As we can see, with the increasing n (1→2→3→4) for n-gram language model, the observed test perplexity decreases gradually. This is because the model has more context to predict the next word. If it has two or three words in your context, it is much easier to predict what the following word will be and thus it lowers the overall perplexity.

>> Sentence Generation:

I have given an input of an incomplete sentence : **"the company"** and tried to generate a sentence of minimum length 10 and maximum length 20.

Based on different models the output (generated sentence and its corresponding probability) that I observed are as follows:

- Unigram:

```
Generated Sentence: <s> the company of to in and said a mln the the the the
the the the the the the </s>
probability: 0.017014244038644752
```

- Bigram:

```
Generated Sentence: <s> the company said it has been made a share </s>
probability: 0.052618529404116515
```

- Trigram:

```
Generated Sentence: <s> <s> the company said it has agreed to sell its
shares in the first quarter of 1986 </s>
probability: 0.02913691081782431
```

- 4-gram:

```
Generated Sentence: <s> <s> <s> the company said it will offer a stake in
the company </s>
probability: 0.032187775883758l4
```

As we can see, with the increasing n, the n-gram model learns better context and predicts more structured sentence . for unigram model, there is no specific meaning of the generated sentence. However, from bigram, trigram and 4-gram we see here is significant improvement and an close to correct or correct sentence structure is generated by the models with some semantic and syntactic meaning.

Neural Network Based Language Model:

For this part of the assignment I have used an LSTM based Reccurent Neural Network to make my language model. During initialisation, I have used pretrained Word2Vec embeddings of vector_size=128 for initialization of the model.

Here I have trained the model on the 90% split of training dataset and the other 10% split was used to finetune the model.

The following set of hyperparameters were tuned during training the model:

1) hidden_size of lstm cell, 2) batch_size, 3) sequence_length,
4) learning_rate, 5) epochs, 6)number_of_layers in lstm cell.

Finally the best finetuned model was obtained for the following values of the hyperparameters:

```
# Hyper-parameters (Obtained after tuning them)
hyperparams = {
    "embed_size" : 128,
    "hidden_size" : 1024,
    "num_layers" : 1,
    "num_epochs" : 5,
    "batch_size" : 20,
    "seq_length" : 30,
    "learning_rate" : 0.001,
}
```

Trained and finetuned Model link:
https://drive.google.com/drive/folders/1HSwCe6Tf729VcQ4LnMcVMLb9h07iHoOw?usp=sharing

Thereafter the average perplexity is calculated on the test dataset using the fine tuned model

The average test set perplexity: **154.42289450497304**

Reasoning:

One question that might come to our mind is that why the average perplexity observed by the neural network is larger compared to the n-gram language model

Neural networks have a few strengths that n-gram models don't have. They can leverage longer word histories, assuming the use of a recurrent neural network. They can also share parameters across similar n-grams.

However, the performance of the models depends on the specific data that we have. As we know, n-gram models are based on counting the probability of observing each possible n-grams. This is a really efficient way to make use of the data especially when you don't have a lot of text to train from. N-gram models can easily beat neural network models on small datasets.

Here we have a training set consisting of 60000 sentences and a test set of 15000 sentences consisting of approximately 44.6k unique words in the training set due to which the n-gram model performs way better than the neural network as n increases. For the unigram model, the neural network easily beats it due to its self learning property, however, for n= 2/3/4, the n-gram model becomes much stronger due to its context specific learning.