E1246 - Natural Language Understanding
Assignment2 : LSTM Language Models

Sachin Mittal (SR Number 14539)

Abstract

In this assignment,I designed a language model on **Gutenberg**(D2) corpus using Long Short Term Memory (LSTM) which was implemented in Python library Keras. Due to time and resources constraints, I trained model on 5 files of Gutenberg Dataset. After training, Model has capacity to predict next word for given few words.

- **Task 1:** Built Token level LSTM-based language model for task-1 :-
  In this task main objective was to build a model such that it predicts next word with given few words, model should remember past history (context) to generate next word, and that makes RNN to use as obvious choice. More specifically, we were asked to implement this model using LSTM.
- **Task 2:** Built Character level LSTM-based language model for task-2 :-
  Here We are asked to implement slightly different variant of above model and that is called Character level LM. It basically predicts the next **character** given few characters.

  one of the fundamental differences between the word level and character level models is in the number of parameters the LSTM has to access during the training and test. The smaller is the input and output layer of LSTM, the larger needs to be the fully connected hidden layer, which makes the training of the model expensive.

1 Token based LanguageModel

- Preprocessing:-
  First I removed punctuation marks from corpus and then sequences of length $50+1$ were formed. Which says that we can feed 50 words at a time into input layer of model, and then model will predict 51^{st} word.

- Model Desciption:-
  To train model adam optimiser in Keras and cross entropy loss were used. Model was trained on 5 files with batch size of 128 over 30 epochs.

2 Character based LanguageModel

LSTM based character-level language models are extremely useful for modeling out-of-vocabulary (OOV) words by nature. To build this model, following two steps were performed.

- Preprocessing:-
  This part remains same as above with the only difference that, this time I created sequences of characters (instead of words) of 51 length.

- Model Design:-
  To train model RMSprop optimiser in Keras and cross entropy loss were used. Model was trained on 5 files with batch size of 128 over 100 epochs.

3 Evaluation Metric

I have used the intrinsic measure of **perplexity** to evaluate the model. The perplexity is given by:

$$PP(W) = P(w_1, w_2..., w_N)^{-\frac{1}{N}}$$

As mentioned above, cross entropy loss was used to train the model, therefore perplexity in code was

implemented by calculating the mean of the cross-entropy loss and exponentiating it as this gives the same results.

4 Result

Perplexity of first task (token level LSTM) over 3 files of Gutenburg were find to be 221.98
Perplexity of second task (character level LSTM) over 3 files of Gutenburg were find to be 180

5 Sentence Genration

We generated sentences using both models. To generate sentence we give some sentence of words or sentence of characters (depending upon model) to input layer of model and then trained model starts predicting one immediate next word of the given sequence and it keeps on doing this to generate desired length sentence.
I have listed out few examples of generated sentences from both models.
Model 1: Token based LSTM LM

- and the lord spake unto moses saying thus saith the
- and the lord shall be the son of man that

Model 2: Character based LSTM LM

- it was very good harriet, the letter. they were all the expected him to be allow the person of mr
- her it would not be at the choilfrioned to have always said, and had let it
- ”i think it was so long was so less before it had been to her to be giving more superior of his f

6 github link

https://git.io/vxgvs