E1246 - Natural Language Understanding
Assignment2 : Neural Language Models

Nidhi Kumari (SR Number 15127)

Abstract

Purpose of this assignment is to build Neural Network based Language model on **Gutenberg** corpus. There are essentially 3 tasks to be performed.

Task 1 :- Token level LSTM-based language model
TAsk 2 :- Character level LSTM-based language model
Task 3 :- Generate a sentence of 10 tokens

1 Token based LanguageModel

- Preprocessing:-

  1. Data in each divided into train set, heldout set and test set in the ratio 8:1:1 based on the number of sentences.

  2. All words are normalized to be in lowercase

  3. Sequences are formed out of that sentences

  4.Each sequence is represented as vector

- Files used from Gutenburg:-

  Text files from gutenberg corpus used are (whole data need more hours)
  1. whitman-leaves.txt
  2. shakespeare-caesar.txt
  3. shakespeare-hamlet.txt
  4. shakespeare-macbeth.txt

- Architecture:-
  1. First I used Embedding layer to represent word in vector form
  2. Two LSTM hidden layers were used with 100 memory cells each. More memory cells and a deeper network may achieve better results. A dense fully connected layer with 100 neurons connects to the LSTM hidden layers to interpret the features extracted from the sequence.

  3. The output layer predicts the next word as a single vector the size of the vocabulary with a probability for each word in the vocabulary. A softmax activation function is used to ensure the outputs have the characteristics of normalized probabilities.

2 Character based LanguageModell

- Preprocessing:-

  1. Data in each divided into train set, heldout set and test set in the ratio 8:1:1 based on the number of sentences.

  2. All words are normalized to be in lowercase

  3. Each character is converted to a number holding position in char-vocab using tokeniser.

  4.Each sequence is represented as vector

- Files used from Gutenburg:-
  For this task also same files were used to make comparison useful.

- Architecture:-
  1. The model is defined with an input layer that takes sequences that have 82 features for the one hot encoded input sequences.
  2. LSTM hidden layers were used.
  3. The model has a fully connected output layer that outputs one vector with a probability distribution across all characters in the vocabulary. A softmax activation function is

used on the output layer to ensure the output has the properties of a probability distribution.

3 Perplexity

Formula of perplexity is given below

$$2^{-\sum_x p(x) \log_2 p(x)}$$

which could also be written as

$$\exp\left(\sum_x p(x) \log_e \frac{1}{p(x)}\right)$$

here $\sum_x p(x) \log_e \frac{1}{p(x)}$ is cross entropy loss, this formula suggests that If we have cross entropy loss then by exponenting it we get perplexity. And thats what I did in my code to calculate perplexity.

4 Result

- Perplexity of task1 is 219.20
- Perplexity of task2 is 187.6

5 Sentence Genration

Sentences are generated with some random input seed and then our model predicts next subsequent words.
below is list of sentences that are being generated using both models-
Token based LSTM LM

- fathers and the lord shall be the son of israel
- and the king of babylon came to the house of

Character based LSTM LM

- she had talking about the character with mr. kn
- her and he said unto him i will not be

6 github link

https://goo.gl/1Hdk1H