BBM497: Introduction to Natural Language Processing Lab. **(Due: 30/03/2020)**

Submission Assignment #1

Instructor: Burcu CAN - Necva BÖLÜCÜ *Name:* Okan ALAN, *Student No:* 21526638

1 Introduction

In this report I discuss my approaches to given tasks, data structures that I am used to my language model, and analysis of errors and results.

2 Dataset

The dataset is too understandable. I examined it easily and changed somethings on a dataset. In the below two-paragraph are the preprocessing steps that I did on the dataset.

When I start to examine the given dataset, I realize that there is gap(empty line) each twenty-first line. After each gap line our stories are shifting 3-4 lines. For example, the first 21 line is like that L1,L2,L3.....,L21 and the second 21 line like that L4,L5,L6.....,L24 these are coming from the big story(L1,L2,L3.....LN , N≥24). Before each these gap lines there is "XXXX" word and end of that line there are consecutive few words such as "*tt yy|tt|xx|zz*". Pobably this dataset prepared for filling the "XXXX" and the "*tt*" is answer. How could I know it? I said our sentences are shifting. When I look at the next shifted part I catch that. Therefore I replaced "XXXX" with the "tt". Lastly, I deleted the "*yy|tt|xx|zz*" from dataset.

Another thing about the dataset is the punctuation mark. When we received each punctuation mark was divided. I deleted some of them such as " “, ‘ , ” , ’ , ? , ! , ” , ; , : , . , –" Some of them are not deleted such as " 're, 's , n't , 've ". I connected them with previous word. There is a sentence in dataset such as "for the king 's aunts were old-fashioned". I tied up "king" and "'s". I think this helped my model to generate more meaningful sentences

3 Language Model

Firstly, when the sentences come to my model, I add beginning token "<s>" and ending token "</s>" to the start and end part of sentence. The count of tokens is N-1 that N is changed according to your model. N is 1 for unigram, 2 for bigram, 3 for trigram and so on.

I am using a dictionary structure to store the word(s). The dictionary became from "string:list" items. The list consists of "dictionary and integer". The big dictionary's key(string) is a previous word(s). When we look at the nested dictionary that is the first element of value(list) of the biggest dictionary, it becomes from the next word with its count. The count is how many times the next word comes consecutively previous word(s). Then the second element of the list is the count of the previous word(s) in the dataset. Let's say this "xxx yyy zzz" is a sentence from the dataset and our model is trigram. If I sent this to the language model. I store it as

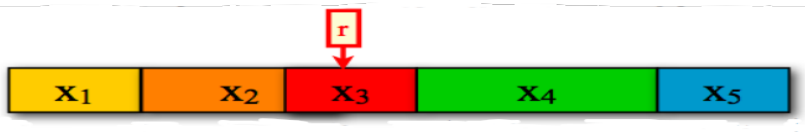
{"xxx yyy" : [{ "zzz" : 1}, 1]}

You can ask what happened the next word in the nested dictionary if our language model is unigram. At that time, I don't store the next word, only I increase the second item of the list.

3.1 Sentence Generation

This part is about the my "Next" function. This function determines the next word coming from taken word(s). Let's look at the how it works

1. Generate the random r number between 1 and X. If our language model is unigram, X is the total count of all words in the dataset. If it is not unigram, X is how many times the previous word(s) appeared in the dataset.
2. Find where r refers



Why I am not using probabilties of words? Because, denominator is same for each related word. Lets examine the trigram language model with following dictionary { "it was":[{ "a": c1, "not":c2, "that":c3, "always":c4, }, c1+c2+c3+c4] }. X which is from first step is c1+c2+c3+c4 because denominator of probabilities are same. Let's look at the formula

$$P(W_k|W_{k-1}W_{k-2}) = \frac{C(W_{k-1}W_{k-2}, W_k)}{C(W_{k-1}W_{k-2})} \tag{3.1}$$

$C(W_{k-1}W_{k-2})$ is equal to X and is same for any next word which will come after W_k. Therefore I thought I can only use the numerator part because floating points confuse my mind.

There is almost same idea for unigram. We can calculate the total count of words with summing the second element of list.

4 Error's In Calculation Perplexity

I use below formula to calculate perplexity. It is the second formula at the assignment pdf.

$$PP(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_1 w_2 \dots w_N)}} \tag{4.1}$$

Our sentences are interrupted because they reach the desired sentence length and I have to finish generating a sentence.Let's say generated sentence finish with "history" but in the dataset there is no sentence which is ending with "history". Therefore denominator is going to be zero and computer breaks itself due to division zero. $P(w_1 w_2 \dots w_N)$ is a probability of sentence because P("¡/s¿"—"xx history") is zero. This problem is occurred when our language model is not an unigram.

5 Results

My results in sentence generation aren't surprising. Unigram language model generate completely worthless sentences because all word are selected ramdomly. There are some examples from my language model generated sentences.

5.1 Unigram Language Model

- 1. Sentence : off am judging night fountains but again janangir but it you bargain a head at report don't little furniture
  - Probabilty of sentence :0.00000000000000000000
  - Perplexity of sentence :2065.98019307524373289198

5.2 Bigram Language Model

- 1. Sentence : strickly speaking of badakhsham in our gift of town dug a long nails and listened in hot rolls blazing fire
  - Probabilty of sentence :0.00000000000000000000
  - Perplexity of sentence :162.17343048021740514741
- 2. Sentence : she could give me
  - Probabilty of sentence :0.00000001089047767183
  - Perplexity of sentence :97.88998630824110591675

5.3 Trigram Language Model

- 1. Sentence : here he is sleeping for fear that chilled his heart for how could it have been over there with her
  - Probabilty of sentence :0.00000000000000000000
  - Perplexity of sentence :19.388193155010736035 14
- 2. Sentence : she did not enjoy anything is blessed which comes out at one mouthful as i wore it next a girdle
  - Probabilty of sentence :0.00000000000000000000
  - Perplexity of sentence :13.238822992778157328 08

If we analyze the perplexity value of the sentences, it goes down when the count of the looked previous word is increased. My code is dynamic. I can create a 10grams mode and more. When I tried, I saw amazing sentences. And the perplexity of those sentences was near to 1. My language model is working.