BBM497: Introduction to Natural Language Processing Lab. **(Due: 02/05/2020)**

Submission Assignment #3

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

1 Introduction

In this report, I discuss my approaches to given tasks and data structures that I used. This report containing how I deal with the given task.

In this assignment, our goal was to learn the how to implement an n-gram(bigram) level Feed-Forward Neural Network(FNN) for Text Generation by using DyNet deep learning library. I think that understanding the structure of the DyNet was the most difficult part of this assignment.

2 Data Preparation

We had two different files. One of them is "glove.6B.50d.txt" that consist of word embeddings. These vectors are used to convert poems into numerical. Other one is "unim_poem.json" that is used to train our neural network. Some words in poems is not our word vector space. So I calculated the mean of all vectors to assign for unknown words. I replaced a few punctuations with white-space. These are " . , : , ? ". Additionally, I replaced "\n" with "eol" token. Then I added "bos" to beginning of the poem and "eos" to end of poem. There were a few tokens to specify the location of sentences or poems when computer learn new things. These are:

bos – begin of sentence
eos – end of sentence
eol – end of line
bol – begin of line

3 Feed-Forward Neural Network(FNN)

In this section, I am going to explain how I build FNN by using DyNet. DyNet is a neural network library developed by Carnegie Mellon University and many others. It is written in C++ (with bindings in Python) and is designed to be efficient when run on either CPU or GPU, and to work well with networks that have dynamic structures that change for every training instance.

The feed forward neural network(FNN) was the first and simplest type of artificial neural network devised. In this network, the information moves in only one direction, forward, from the input nodes, through the hidden nodes (if any) and to the output nodes. There are no cycles or loops in the network.

In the assignment you wanted us bigram model but I coded dynamic gram. It means I can change my model from bigram to trigram with changing only one input parameter. Below figure1 is trigram model.

I used a dictionary to store n-gram model word frequency that will use on calculating perplexity. Its structure looks like { PrevWord(s) : ["NextWord":cc, tt] } such as { "be" : ["or":1, "eol":1,2] }. This is from below figure's 1 bigram version.

3.1 Training

Training takes a long time that's why I used only a small part of the poems 5% of all. Lets examine my model: input comes as (50,1) vectors, input layer contains 150 nodes, also hidden layer contains 150 nodes and the softmax function is running in the output layer and produce a probability list as large as the number of unique words. Then I was updating the weights between nodes.

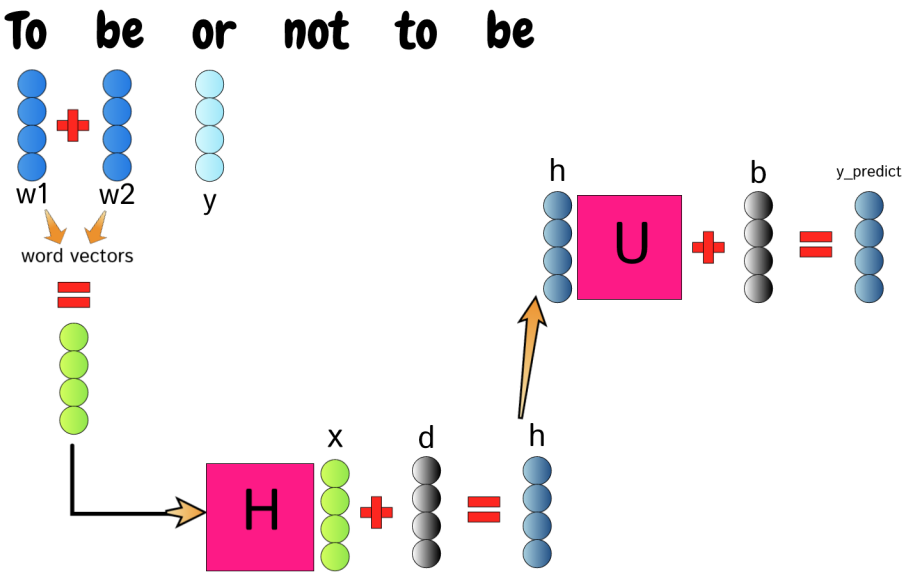

Figure 1: FNN model

4 Generate Poem

We will produce poetry using the models we have just trained. How is it possible? You know the softmax function generates a probability list of the next word. These are used for creating a probability chart to determine the next word.

4.1 Examples

- Poem 1:
  ascending resonant have end grip shall we occurred blue
  each breathy cheek end wet thrones hiding adjusted kind
  share darkness only choice my blood return
  Perplexity : 224.8268188472322322013

- Poem 2:
  undone the share law been me heaven freight don't
  undone freight up saw darkness uncontrolled undone adjusted delight
  undone staring life staring woman's undone milne
  Perplexity : 225.321654745633334333

- Poem 3:
  rarely sails if share freight up law the staring
  rarely sails life the moonlight rarely freight me
  feature undone the love at staring life at
  law came resonant undone missing' shall frail o bribe
  Perplexity : 317.98235525646435917224

- Poem 4:
  rule me big adjusted
  up beside patiently
  Perplexity : 36.99427219711688508141 Perplexity :

- Poem 5:
  Perplexity :

5 Pseudo-Code

1. Read poem dataset
2. Apply preprocessing to poems
3. Read Glove(word embeddings)
4. Train Feed-Forward Neural Network
5. Generate poem
6. Calculate perplexity of generated poems