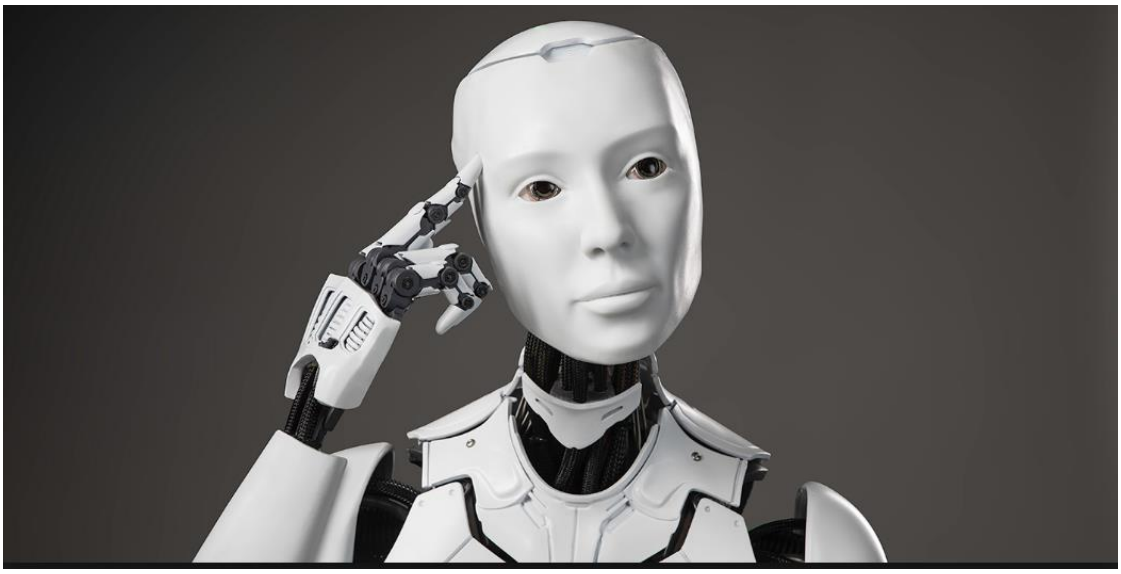
ΔΗΜΟΚΡΙΤΟΣ
ΕΘΝΙΚΟ ΚΕΝΤΡΟ ΕΡΕΥΝΑΣ ΦΥΣΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

Chatbot using RNN and Tensorflow



Avgeros Giannis 2022201704003

Msc Data Science

Athens 2019

Introduction

The purpose of this assignment was to create a deep RNN that will serve as the chatbot. Chatbots handle incoming questions from a user and, using a dataset and a deep RNN, is trained to respond to the question asked, accordingly.

Due to the enormous number of numerical transaction needed for the model to be trained, it was essential to execute the jupyter notebook in an environment that would contain a GPU to make use of the extreme optimization the recent video graphic cards present. Thus, by using my gmail account I created and tested my program in Google Collaboratory, a free and open jupyter notebook with tensorflow that uses a provided gpu by google. The jupyter contains the code supposed to be run in a gpu, and it also contains the preprocess of the dataset which I explain in detail there, thus i will not cover preprocess steps in this report.

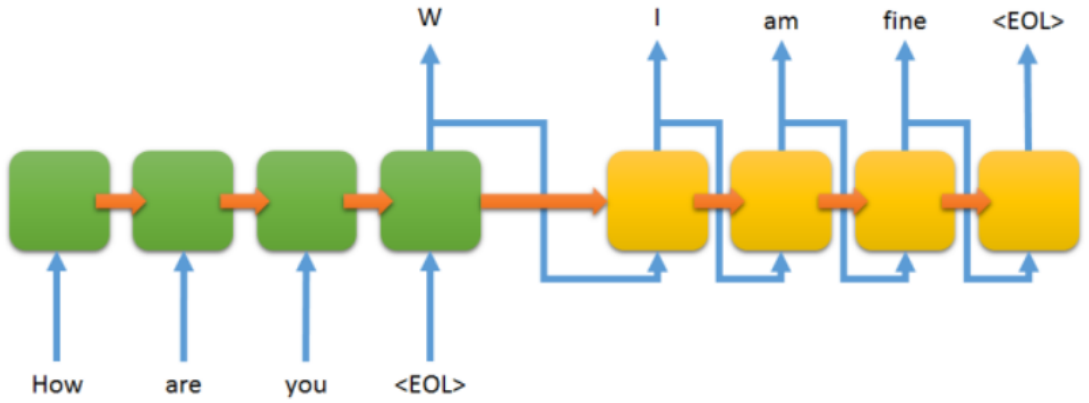
Recurrent Neural Network Architecture

Firstly, small introduction to recurrent neural networks(RNN). They are a distinct category of neural networks that can deal with sequential data (videos, text sequences etc). The network tries to understand the meaning of the sequence by translating the sentences to vector of numbers and identifying key characteristics and connection between words (to predict a word in a sequence, you need to know what was the previous words) using the structure and the position of the symbols. What separates them from the remaining neural networks is that it contains an internal state, which servers the purpose of maintaining a (vanishing) memory about history of all past inputs.

There are three types of RNNs: Vanilla, Long-Short Term Memory (LSTM) and Gated Recurrent Units(GRU'S). Vanilla is the simplest of all, but faces the problem of vanishing gradients (as better explained by Wikipedia "*In such methods, each of the neural network's weights receives an update proportional to the partial derivative of the error function with respect to the current weight in each iteration of training. The problem is that in some cases, the gradient will be vanishingly small, effectively preventing the weight from changing its value and, finally, stop further training way sooner than should")*. From the remaining two, I decided to use the LSTM because, due to the fact that may be slower than the GRU when training, it seems that it has better results for larger and more complex sequences(see *Comparative Study of CNN and RNN for Natural Language Processing*).

I decided to use the tensorflow seq2seq implementation. Sequence-to-sequence is a model, consisted of two RNNs - an encoder and a decoder. The encoder reads the input sequence, word by word and emits a context (when the encoder reaches the end of the sentence it emits the final internal state), which is the understanding of the network for the meaning of the sentence. Based on this context, the decoder generates the output sequence, one word at a time while looking at the context and the previous word during each timestep. The decoding model can be thought of two separate processes, training and inference. It is not they have different architecture, but they share the same architecture and its parameters. It is that they have different strategy to feed the shared model. While training phase, the input is provided as target label, but they still need to be embedded. On the inference phase, however, the output of each time step will be the input for the next

time step.

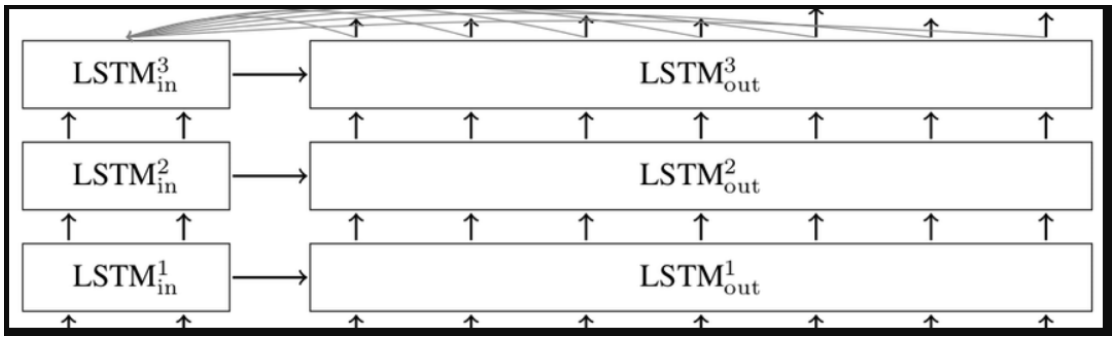


Εικόνα 1 The input sequence is "How are you". When such an input sequence is passed though the encoder-decoder network consisting of LSTM blocks (a type of RNN architecture), the decoder generates words one by one in each time step of the decoder's iteration.

Architecture Implementation

As far as the main architecture implementation, I decided to use a 3 layer seq2seq implementation, meaning there will be 3 consecutive layers of encoder-decode (because "*deep LSTM vastly outperform shallow LSTM*")[1] , and each layer will have the same amount of hidden units as the size of the embedding. In the end of this model, there is a Dense Layer. The output of a LSTM is not a softmax. Dense layer is used as a probability distribution estimator, it produces a vector with size equal to the size of the vocabulary and it produces the possibility of a word to be the selected one based on the previous words and the internal state of the previous layer.

[1]: Sequence to Sequence Learning



Εικόνα 2 The 3 layer model of LSTM encoder and decoder. The lines in the last lstm out going back to lstm in is a mechanism called attention NOT included in our implementation.

For the two networks before mentioned, values must be set to two parameters: embedding size and vocabulary size. Vocabulary size is the size of the distinct words in question and answer, a calculated parameter. Embeddings are the vector representation for words that aims to identify similarity between them by calculating the distance between their vectors. For the embedding size, I decided to set the size of it to be 512 (lower values

did not produce equally good results/higher values made the training time insufferably longer). I also decided to split the dataset into batches of 128 elements.

Loss Function

For the loss function, I used the cross_entropy_seq_with_mask. I use the cross entropy loss function (cross-entropy describes the loss between two probability distributions and measures how close is the predicted distribution to the true distribution), alongside mask. Mask is a vector containing 1's where the sentence have a word and 0's when it has PAD item. The mask is going to be used to skip any input with mask 0 by copying the previous hidden state of the cell; it will proceed normally for any input with mask 1. Masking allows us to handle various length inputs in RNNs.

For the training phase I decided to use 25 epochs. The loss value keeps on becoming smaller and smaller, as seen:

Epoch [1/25]: loss 5.1797
Epoch [2/25]: loss 4.6344
Epoch [3/25]: loss 4.4848
Epoch [4/25]: loss 4.3925
Epoch [5/25]: loss 4.3208
Epoch [6/25]: loss 4.2625
Epoch [7/25]: loss 4.2111
Epoch [8/25]: loss 4.1669
Epoch [9/25]: loss 4.1270
Epoch [10/25]: loss 4.0900
Epoch [11/25]: loss 4.0556
Epoch [12/25]: loss 4.0256
Epoch [13/25]: loss 3.9961
Epoch [14/25]: loss 3.9687
Epoch [15/25]: loss 3.943
Epoch [16/25]: loss 3.9189
Epoch [17/25]: loss 3.8973
Epoch [18/25]: loss 3.8751
Epoch [19/25]: loss 3.8553
Epoch [20/25]: loss 3.8361
Epoch [21/25]: loss 3.8192
Epoch [22/25]: loss 3.8015
Epoch [23/25]: loss 3.7854
Epoch [24/25]: loss 3.7706
Epoch [25/25]: loss 3.7545

Εικόνα 3 Loss function compared to the epochs

Conversation example

Below, I present the results of two conversation's I initiated with the chatbot:

```
================================================================================

-how are you
- oh fine i am fine i am fine

-i love basketball
- what do you mean

-do you love sports
- i am sure

-obama won
- i am not sure i have been <UNK> for a while

-updated?
- you are a good man

-do you drink and drive
- no i do not

-why you do not?
- because i do not know

-how to drive?
- it is not a game

-for me it is a game
- i will not be needing it


-who cares about you
- you know what you mean

-you are arrogant
- i am not

-sometimes you are
- i am fine

-you wish you were
- no i did not

-because you suck big time
- i am not
```

```
-This is such a pleasant day.
- yeah it is

-i want to travel the world
- you do not know

-i sure do know
- you do not have to do it
```

```
-but i really like france
- no you do not

-i really like greece then
- you do not know what you are talking about

-i want to see a basketball game live
- you are a <UNK> <UNK> and i am not going to be here

-sometimes i feel lonely
- i know

-do you generally feel that to?
- i do not know

-you do not feel
- no i do not

-i love the way we talk
- what do you want

-to know you better
- what is it

-how about a coffee
- i will be right back

-that must be a negative response
- i am sorry i am sorry
```

==========

And a third one, where the word we chose would be selected out of the top words with the highest probability. That indicates that the answers we will get will vary more .

```
-my heart hurts
- i am sorry

-i feel negligible
- and the next time you are going to get a lot more

-i feel empty
- well what happened

-i lost my mother
- what is it

-the person who gave birth to me
- you will have to <UNK> it up

-suck ?
- i will be back soon

-why you always leving
- i am an actor

-what do you do
- i am going to kill her
```

-Who?
- your wife she is a <UNK>

-What the hell
- what are you talking about

-Robots really to0k over
- and what are you thinking about

-stopping you all
- and you are the one who wants to be a fireman

-i never said that
- i have heard what i wanted