Chatbot Implementation using Seq2Seq and Transformers

Yanming Liu

Krishna Hemant

liu.yanmi@northeastern.edu

durgasharan.k@northeastern.edu

Signature of Student 1: *Yanming Liu*

Signature of Student 2*: Krishna Hemant*

Submission Date: 06/12/2022

Abstract

This paper studies and implements two neural network models , *Seq2Seq* and *Transformers* to build a Chatbot using the Cornell Movie dataset and finds the best Hyperparameters for the model to produce a Human-Like chatbot and would further explore the models, evaluate and discuss why the Transformer is better and efficient model and how it could lead to an exciting future.

Introduction

As computation algorithms get more powerful everyday, companies are starting to rely on machine learning models to lend a data driven helping hand in solving basic but important day to day activities which involve automation. The Chatbot, or a fancier term - assistant, has started to make a big appearance into everyday lives through assistants like Siri or Cortana, or in automating customer service for companies. Retaining the workforce for customer service roles has been a huge task for companies as people quit due to burnout and routine-work. A Chatbot can have a huge impact in the future of companies and assistants as they can handle customers in parallel and have more efficient and optimized communication which can help people as well the company in increasing productivity and Revenue. This paper deals with understanding and implementation of the chatbot using specific Neural Network architectures which can aid the world to a better future.

The data was taken from the cornell website
https://www.cs.cornell.edu/~cristian/Cornell_Movie-Dialogs_Corpus.html

Background

Chatbot has excellent potential in assisting individuals on tasks such as ordering a movie ticket or having a companion to talk to. [1]. The previous method for constructing a chatbot mainly focuses on rules or retrieval that are unable to handle a full conversation and are less human-like. In contrast, the generative approach incorporated with deep neural networks provides the possibility for an open-domain chatbot for a sensible, long-lasting, and empathetic response [2]. Researchers have built a Seq2Seq model with LSTM and Gated Recurrent Units (GRUs) with an attention mechanism, [3-4]. In addition, less training time was present with the Transformer encoder-decoder attention model than LSTM one for Automation Speech Recognition (ASR) [5].

Approach

Data Description

This corpus contains a large metadata-rich collection of fictional conversations extracted from raw movie scripts:

This corpus contains 220,579 conversational exchanges between 10,292 pairs of movie characters,involving 9,035 characters from 617 movies, having 304,713 utterances in total.It also contains movie metadata like IMDB ratings and character names but we will ignore that and focus on the conversational exchanges to train our chatbot

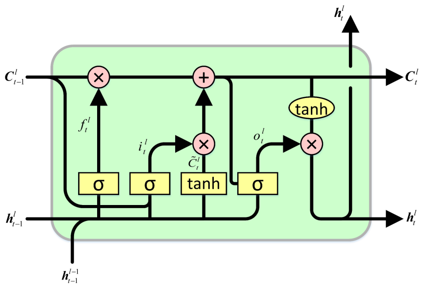
Data Cleaning and preparation

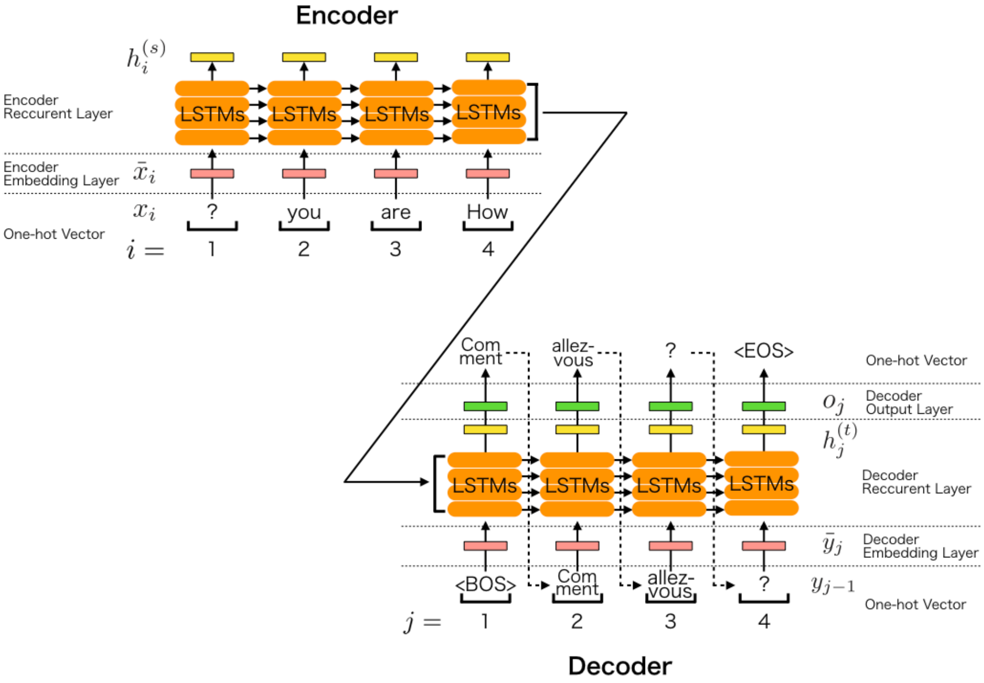
- 50,000 samples were selected for training, validation and testing and were split in the ratio 7:2:1. Maximum length of 60 words were selected for question and answer pairs (Transformers) and maximum character length of 20 was selected for our seq2seq model.
- Our selected data was entirely converted to lowercase and we applied regex functions to take care of punctuations and special characters. A space “ “ was added between words and punctuations for ease of word extraction

Here is a general introduction to the models we have used with helpful illustrations,

Seq2Seq (LTSM)

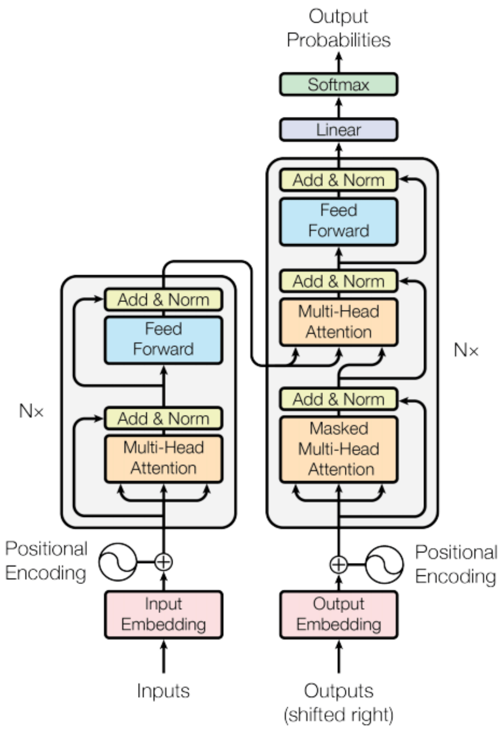
We used this Neural Network architecture encompassing a encoder and a decoder which is used to predict an output sequence from an Input sequence, it is also used for language translation tasks. It uses multiple RNN’s which are stacked so that they transfer their hidden states along the model. The Seq2Seq model is better than the naive RNN model as it does not predict the results immediately and takes in account the whole sentence and passes a summarized vector to the decoder and that decodes into a response sentence. The model was implemented with the help of a textbook [6] .Here is an illustration of the Seq2Seq model architechture and an LSTM cell used in the model.

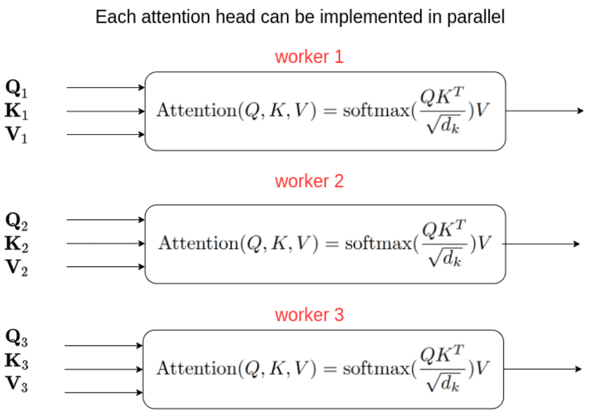




The Transformer

The transformer is a breakthrough model which was released in December 2017 by Vaswani et al. in their paper *Attention is all you need,* we can take this literally as we implemented the transformer model with its attention mechanism to get great results with our limited GPU resources. The transformers textbook [7] was very helpful in revision and implementation of the model. It also encompasses a encoder and decoder model similar to a seq2seq, but with a few sublayers and the important attention mechanism, the images below will help to give a better understanding.




Each attention head can be implemented in parallel
worker 1
$\mathbf{Q}_1$
$\mathbf{K}_1$
$\mathbf{V}_1$
$\mathrm{Attention}(Q,K,V) = \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V$
worker 2
$\mathbf{Q}_2$
$\mathbf{K}_2$
$\mathbf{V}_2$
$\mathrm{Attention}(Q,K,V) = \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V$
worker 3
$\mathbf{Q}_3$
$\mathbf{K}_3$
$\mathbf{V}_3$
$\mathrm{Attention}(Q,K,V) = \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V$

Model Implementation and Results

In this section we will discuss the models implemented and the results obtained in detail with parameters used and graphical results based on model performance

Seq2Seq

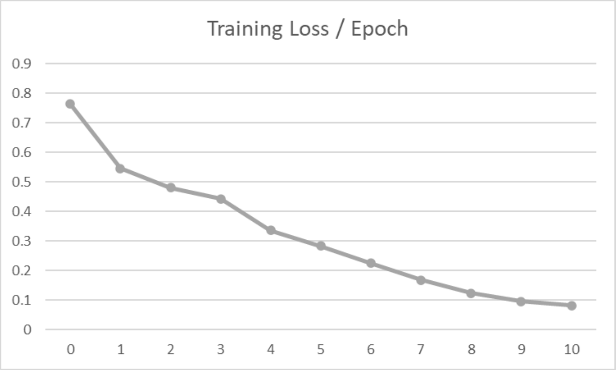
We implement the Seq2Seq (sequence to sequence model) with RNN's using the LSTM mechanism which helps with the vanishing gradient problem. The specifics of the model including the implementation environment and hyperparameters used to train are given below

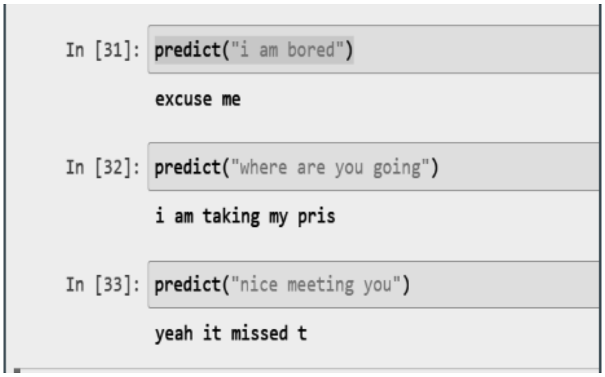
Hardware: Intel core i5-8250 @1.60 GHz, 16 GB ram, NVIDIA Geforce 940mx
Language and version: Python 3.6 + tensorflow 1.14.0 ,py36 env
Hyperparameters:

BATCH_SIZE	NUM_LAYERS	D_MODEL	Embed_Dim	Learning Rate	EPOCHS
32	2	512	64	0.0003	10

Evaluation and prediction results:

Our seq2seq learns from the training dataset as the training loss decreases from around 0.78 to 0.09 when epoch increases. Moving to the prediction response in the right graph below, it has some meaning related to the question, however, it is not good enough. The reason why the chatbot stops at a certain length of response is that we set a limited character length for not generating sentences. We use this as a baseline to compare with the performance of Transformer.





```
In [31]: predict("i am bored")

         excuse me

In [32]: predict("where are you going")

         i am taking my pris

In [33]: predict("nice meeting you")

         yeah it missed t
```

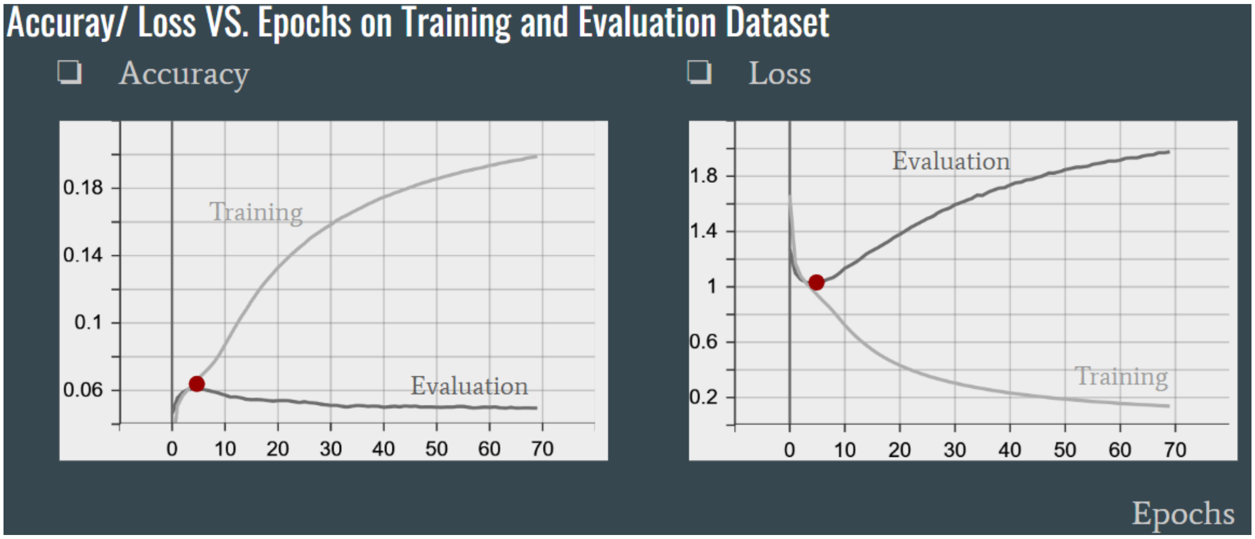
<u>**Transformers**</u>

We implemented and tested our transformers under conditions below,

Hardware: Google colab pro Gpu: Tesla T4
Language and version: Python 3.8 + Tensorflow 2.9.2
Hyperparameters:

BATCH_SIZE	NUM_LAYERS	D_MODEL	NUM_HEADS	UNITS	DROPOUT	EPOCHS
64	2	256	8	512	0.1	70

To highlight, the model was trained on the training dataset for 70 epochs, evaluated on the evaluation dataset using the weights per epoch learned, and tested on the test data via metrics like accuracy and loss. The graph below illustrates that the model is ‘acquiring knowledge’ along epochs as the training accuracy grows or the training loss decreases. However, when the model is faced with the unknown evaluation dataset, it performs well during the first approximate 5 epochs (position at red point), then reaches the peak well-behaved point, followed by downward trend in evaluation accuracy. That is true of training loss in intuition. This phenomenon signify that the existence of overfitting, which shed a light on the further hyperparameter tuning on epoch to stop 2~3 epoch after the x-label of red point.



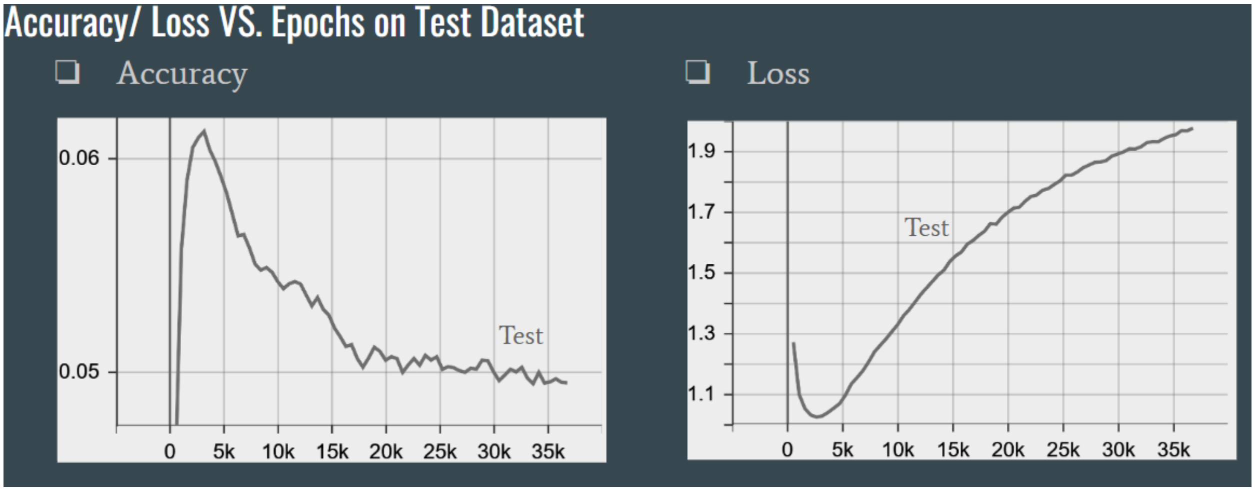
Accuray/ Loss VS. Epochs on Training and Evaluation Dataset

❑ Accuracy

❑ Loss

Epochs

Below is the accuracy and loss on the test data, with a size of 5000. Both metrics share the insights of overfitting crossing certain steps as we mentioned above. Here, I mistakenly use the evaluation dataset with 35k question-answer pairs for testing purposes, however, I will update it before next presentation.

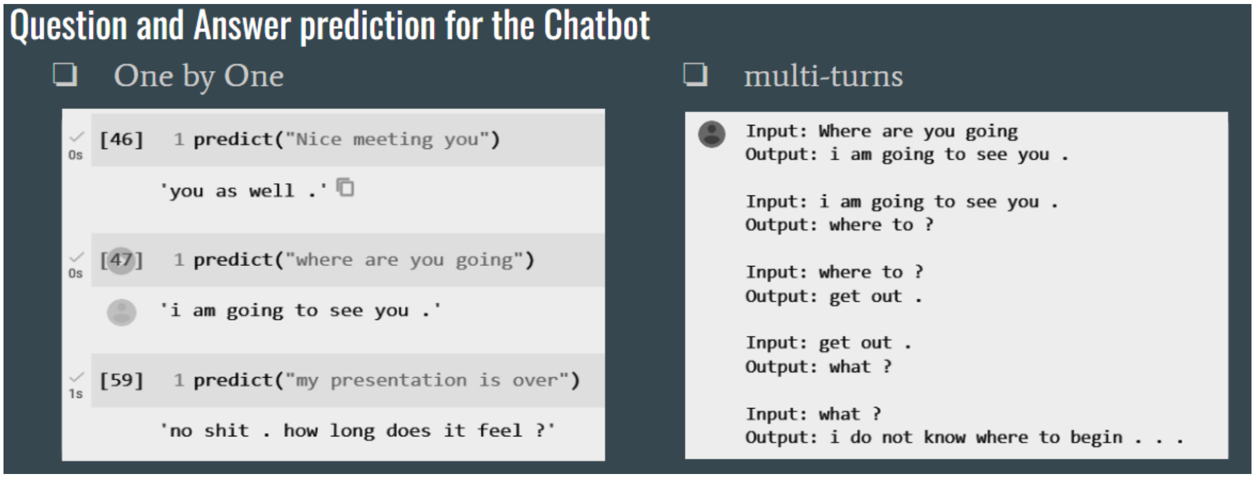


Accuracy/ Loss VS. Epochs on Test Dataset

❑ Accuracy

❑ Loss

The response of the chatbot is as follows in one-by-one and multi-turns. They show the empathy meaning, which means that the implementation is ready to be extended for other purposes like parameter tuning.



Question and Answer prediction for the Chatbot

❑ One by One

```
[46] predict("Nice meeting you")
     'you as well .'

[47] predict("where are you going")
     'i am going to see you .'

[59] predict("my presentation is over")
     'no shit . how long does it feel ?'
```

❑ multi-turns

```
Input: Where are you going
Output: i am going to see you .

Input: i am going to see you .
Output: where to ?

Input: where to ?
Output: get out .

Input: get out .
Output: what ?

Input: what ?
Output: i do not know where to begin . . .
```

Discussion

After implementing the chatbots using seq2seq models on cornell movies corpus, we can tell that the transformer is superior to the seq2seq model for the following perspectives. Transformer is faster in training iteration than Seq2seq, for they employ self-attention mechanisms instead of RNN. Parallelization can be processed for the whole input sentence instead of calculating semantic vectors one by one. Furthermore, transformers usually have better responses and are closer to a human-like response. Just as in the paper title, attention is that all you need, it has already encoded the data in a much more intelligent ways compared to the one that seq2seq uses, so that the classification algorithm can easily learn what differs in the data. However, because of the existence of multi-heads in parallelization, the transformer occupies extra memory.

For future work, we may target three parts, hyperparameter tuning, performance metrics and additional current-of-state models. Lacking powerful gpu and familiarity to tensorflow 1 & 2, we are not able to do hyperparameter tuning this time, which constrains our model potentials. Then, BLEU score, quantitative time and memory usage should be introduced in the next trial to better measure our model, and use not only the accuracy or loss or predicted response that we currently tested. Finally, reinforcement learning with the intrinsic nature of learning from the feedback, is a promising generic method for improving the chatbot response and we plan to implement it next.

References

[1] https://arxiv.org/pdf/2111.01414.pdf
[2] https://link.springer.com/chapter/10.1007/978-3-030-49186-431
[3] https://arxiv.org/ftp/arxiv/papers/2006/2006.02767.pdf
[4]https://dl.acm.org/doi/pdf/10.1145/3377713.3377773
[5] https://ieeexplore.ieee.org/document/9004025
[6]https://www.manning.com/books/machine-learning-with-tensorflow-second-edition?query=Chris%20 Mattmann
[7] Transformers for Natural Language Processing: Dennis Rothman

All team members contributed equally