International Journal of Technology Research and Management
ISSN (Online): 2348-9006
Vol 6 Issue 3 March 2019
NCETRCSE-2019 Conference Proceedings, CSE Dept. LNCT, Indore

An Implementation of Deep Learning Based Auto Answering System using Tensor Flow Framework – Chatbot

Harshwardhan Singh[1], Hamendra Umath[2], Dr. Harish Patidar[3]
Computer Science, Lakshmi Narain College of Technology, Indore, 453555, India[1, 2, 3]
hackwidharsh@gmail.com[1], hamendrarajput507@gmail.com[2], harish.patidar@gmail.com[3]

Abstract: *Chatbots are becoming immensely popular with the advancements in the technology. Chatbot is a computer program which could talk to people in real time and give intelligent replies. The easy to use interface and capability to outperform humans with the speed of handling user queries makes it of extreme importance. This paper addresses the design and the implementation of the chatbot with the use of deep learning and TensorFlow models such as Neural Machine Translation. We will also study various areas where chatbots are making a huge impact and the techniques used while designing a chatbot.*

Keywords: *deep learning, python, chatbot, machine learning, neural machine translation.*

1. INTRODUCTION

The last few years have seen a huge boom in the Artificial Intelligence (AI) and Machine Learning industry. One of the major challenge is to build a chatbot which could talk to humans in a natural language but this task is also being accomplished with the use of AI. Chatbot is a computer program which mimics human conversation using Natural Language. Chatbots have the capability to outperform humans with the speed of handling user queries and their easy to use interface. Improved efficiency and round the clock usability makes them of huge importance. In recent times neural network approach has been mainstream in the industry because of its learning ability from hidden relationships in data without imposing any fixed relationships in the data. A Sequence-to-sequence model accompanied with Attention Mechanism proves to be an astonishing solution for response generation in machine translation.

In this paper we will be studying the problem of responding to user queries with the aim of establishing the conversation in natural language which the user could understand and would expect the response from anyone else. For example if a user asks the bot – "How are you doing?" The bot responds by saying: 'i am doing fine, thanks for asking', which is considered a good response. The data which is used to train the bot is huge and its responses will be accordingly, depending on the amount of training and testing.

The first known chatbot was ELIZA, developed in 1966 which used pattern matching and responded to user queries in the form of questions but its conversational ability was not good enough.[1] ELIZA was the start of the new era and the industry has since then seen improvement on a huge scale. In this paper we have used sequence to sequence approach and used bidirectional neural network to train the bot. This model consists of an encoder and a decoder which assign a word vectors to the input sentence which then passes through thought vector and then the decoder. A Bidirectional Recurrent Neural Network is used in this model.

2. RELATED WORK

There exist a number of models based on sequence-to-sequence framework which work in different fields and generate responses accordingly. For example, Topic Aware Neural Response Generation Model by Chen Xing and others, this leverages on the idea of training the bot by topic information into response generation. Yao et al. (Yao, Zweig, and Peng 2015) added an extra RNN layer in their model between the encoder and the decoder of the sequence-to-sequence model. Li et al. (Li et al. 2016) built a personalized conversation engine by adding personal information as an extra input. In this model we consider training the bot with different training data and note its progress with various models using the sequence-to-sequence for response generation.

3. MODEL

In neural networks, considering a task like sequence to sequence, where the sequences aren't exactly the same length. This is true for chatbots but is also valid for other such tasks. In the case of a chatbot, one word input could yield 20-word output response, and also long statements could end up giving single word responses. Each of these inputs are different from the last in terms of characters, words, and more. Words themselves will be assigned either arbitrary or meaningful id (via word vectors), but handling variable length is a tough task. One answer is to just make all strings of words 40 words long. But then, when the statements are 20 words long, we could just pad the other 10. Any data longer than 40 words, we can either not use for training, or truncate as it can make training hard, especially for shorter responses which might be the most common responses and most of the words/tokens will just be padding. The original sequence-to-sequence example used bucketing to solve this issue, and trained with 4 buckets. 5-10, 10-15, 20-25, and 40-50, and we would just end up putting the training data into the smallest bucket that would fit both input and output, but this is not an ideal approach.

Then we have the Neural Machine Translation (NMT) code that works with variable inputs, no bucketing or padding. Our model contains support for attention mechanism, which are meant for adding longer-term memory to the recurrent neural networks.

4. NEURAL MACHINE TRANSLATION

4.1 Sequence-to-sequence Model

Sequence-to-sequence models have been hugely successful in the task of machine translation. [2] The earlier methods worked by breaking the input sentence into multiple parts and then do the translation phrase by phrase. But this is not how humans communicate. We read the complete input sentence, understand its meaning and then produce its response. A NMT does this task.

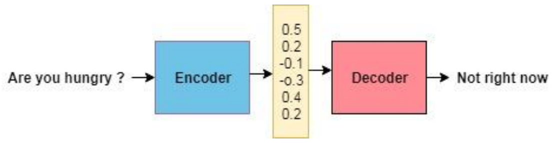

Figure 1: Encoder-decoder architecture

[3] The inner working of encoder consists of reading the input sentence and then it builds a thought vector. Vector is a sequence of numbers which represents the meaning of the sentence. The decoder then processes the sentence vector and produces a response. This is how NMT solves the problem in traditional phase based approach. The objective function of Sequence-to-sequence can be written as:

$$p(y_1, \dots, y_{T'} \mid x_1, \dots, x_T) = p(y_1 \mid c) \prod_{t=2}^{T'} p(y_t \mid c, y_1, \dots, y_{t-1}) \qquad (1)$$

4.2 Attention Mechanism

A Long Short Term Memory (LSTM) can remember sequences of tokens up to 10-20 in length adequately. Any training sentence longer than this limit may drop the performance and the network forgets the initial tokens to make room for the new ones. Here tokens are words which means that a basic LSTM is capable of learning 10-20 word-length sentences. As the sentence length increases further than this, the output response might not be as per the expectations. Attention mechanisms help in generating longer "attention spans," which help a network to reach more like 40-50 words. Apart from increasing sentence lengths the Attention Mechanism also helps in improving the shorter responses which ultimately results in far more complex learning than what we need for a chatbot.

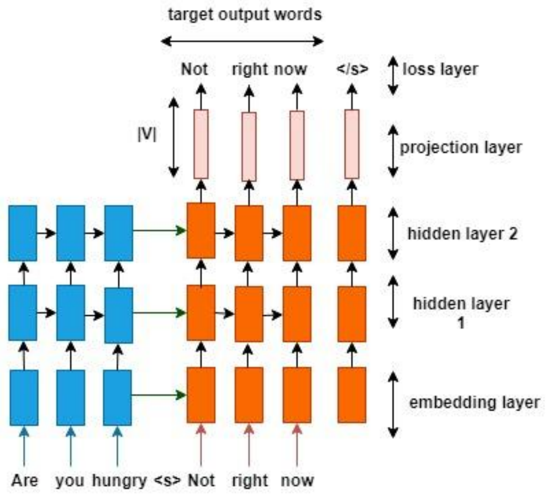

Figure 2: Attention Mechanism

As shown in Figure 2, the attention computation is done at every decoder time step. [4] It consists of the following stages:

1. Current target hidden state is compared to source states to derive attention weights.
2. A context vector is computed based on attention weights as the weighted average of the source states.
3. Combining the context vector with current target hidden state yields the final attention vector
4. The attention vector is then fed as an input to the next time step. The above three steps can be summarized

by below equations:

$$\alpha_{ts} = \frac{\exp(score\ (h_t, \bar{h}_s))}{\sum_{s'=1}^{S} \exp(score\ (h_t, \bar{h}_s))} \text{ [Attention weights]} \qquad (2)$$

$$c_t = \sum_s \alpha_{ts} \bar{h}_s \text{ [Context vector]} \qquad (3)$$

$$\alpha_t = f(c_t, h_t) = \tanh(W_c[c_t : h_t]) \text{ [Attention vector]} \qquad (4)$$

$$\text{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s) = \begin{cases} \boldsymbol{h}_t^\top \boldsymbol{W} \bar{\boldsymbol{h}}_s & \text{[Luong's multiplicative style]} \\ \boldsymbol{v}_a^\top \tanh(\boldsymbol{W_1} \boldsymbol{h}_t + \boldsymbol{W_2} \bar{\boldsymbol{h}}_s) & \text{[Bahdanau's additive style]} \end{cases} \qquad (5)$$

5. EXPERIMENTS

We test various models based on the training data-set size and vocabulary and compare their results.

5.1 Training Data

Training data is used to train an algorithm and generally, training data is a certain percentage of an overall dataset along with the testing set. The performance of the model depends directly upon the quality of the training data, better training data results in a better model. Algorithms learn from data and find relationships, develop understanding, make decisions, and evaluate their relevance from the training data they're given. [5]The better the training data is, the better the model performs. In fact, the quality and quantity of our training data has as much to do with the success of our data project as the algorithms themselves. Now, even if we store a vast amount of well-structured data, it might not be labeled in a way that actually works for training our model. In this project we have trained the bot with different models and noted its response based on the various training and testing data sets. The training data is divided into a comment and response based set. Initially the training set contained redundant information such as 'author', 'time', 'year' and many more. These parameters were not needed and hence the irrelevant data was cleaned and the training model was generated. The chatbot performed differently in every model with its responses.

Table 1: Comparison of Models

Model No	Dataset Pairs	Vocabulary	PPL
1	1000	1000	1100.91
2	2000	2000	2163.28
3	10000	10000	900
4	100000	20000	540
5	2500000	100000	84.68

5.2 Setup

We built our data set from Reddit which is a social news aggregation, web content rating, and discussion website which allows users to post and comment on other people's post. We crawled over 5 million post-comment pairs and used them to simulate message-response pairs in conversation. We removed comments which had no replies, replies which were website addresses, and replies with length greater than 50 were also removed. After this pre-processing, there are around 2.5 million pairs left. From these pairs we select different data-sets for our models which are then used to train the chatbot.

6. EVALUATION METRICS

There exist a number of factors to evaluate the performance of a chatbot. The chatbots differ widely in their inner working and the purpose which they are meant to serve, likewise the evaluation should be done accordingly. A chatbot which is meant for serving customer queries should be evaluated based on the queries which day to day customers are going to ask the bot.

The methods used to evaluate bots are still not very reliable and in the development phase. In this paper we used the following metrics to check the performance of our chatbot:

6.1 Perplexity: Perplexity (PPL) is a measure of how good the responses are generated by the trained model. [6] A lower PPL score suggests that the model is good at predicting the responses. In this paper we used perplexity to determine the performance of the chatbot [7]. Value of PPL is a determining factor as to when to halt the training. Initially the PPL value is very high and it starts decreasing while the model is being trained. We can stop the training once the decrease in PPL value is not very significant. PPL can be defined by the following equation:

$$PPL = exp\left\{-\frac{1}{N}\sum_{i=1}^{N}\log(\text{p}(Y_i))\right\} \qquad (6)$$

6.2 BLEU Score: Bilingual Evaluation Understudy (BLEU) score is also an effective way of determining the effectiveness of our response system. In conversational data there is no exact reply to a particular question. In our system we are translating sequences to sequences which are both in English, so we might not see a very high BLEU score as in the case of language translation tasks. The BLEU score will rise over the time. [8]But the BLUE score is not an ideal evaluation for a chatbot since the human response is more diverse and is not in its scope. [9]The BLEU value and human judgment differs widely and thus it is not considered as an evaluation metric. Therefore, we will be using PPL as an evaluation factor.

7. RESULT ANALYSIS

The performance of the chatbot varies in accordance to the training data set it was trained with. In Model 1 the data set pair was very low and thus the PPL value is also high as the model did not have enough data and the vocabulary to train. The subsequent models i.e. 2 and 3 also did not perform very well as they generated mostly single word and often repeating replies with lots of unknown tokens due to their small vocabulary. Model 4 was trained with 100000 pairs and had a PPL of 240 which is still very high but this version was better than any previous ones. The replies were not single word but still it had repeating vocabulary with unknown tokens. Model 5 was trained with full data set pairs and huge vocabulary of 100000 tokens. This model is significantly better than any previous model and due to the high amount of vocabulary, it doesn't contain unknown tokens and its replies are also better than any of the previous models.

8. CONCLUSION

The performance of the chatbot varies as per the size of the training data set. As the training size and training time increases, the PPL decreases which results in better response generation. The chatbot can be trained as per the need of the customer by using the data set of that particular field which could result in more specific training and satisfactory replies.

ACKNOWLEDGEMENTS

We would like to thank Intel AI DevCloud for providing Machine Learning Frameworks and cloud computation platform for training the chatbot.

REFERENCES

[1] Luka Bradeško and Dunja Mladenić. "A Survey of Chabot Systems through a Loebner Prize Competition", Semantic Scholar, 2012
[2] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. "Sequence to sequence learning with neural networks". NIPS, 2014.
[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate", ICLR, 2015.
[4] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. "Effective approaches to attention-based neural machine translation", EMNLP, 2015.
[5] Yajuan Lü, Jin Huang and Qun Liu, "Improving Statistical Machine Translation Performance by Training Data Selection and Optimization", EMNLP-CoNLL, 2007.
[6] Stanley Chen, Douglas Beeferman, Ronald Rosenfeld, "Evaluation Metrics for Language Models", CMU, 2001.
[7] Rico Sennrich, "Perplexity Minimization for Translation Model Domain Adaptation in Statistical Machine Translation", University of Zurich, 2012.
[8] Elior Sulem, Omri Abend, Ari Rappoport "BLEU is Not Suitable for the Evaluation of Text Simplification", The Hebrew University of Jerusalem, 2018.
[9] Chen Xing, Wei Wu, Yu Wu, Jie Liu, Yalou Huang, Ming Zhou and Wei-Ying Ma., "Topic Aware Neural Response Generation", Cornell University, 2017.