Email Autoresponder using a Sequence-to-sequence Architecture with BERT

Claudio Scheer and **Jose Fernando Possebon Junior**
Pontifical Catholic University of Rio Grande do Sul - PUCRS
{claudio.scheer, jose.possebon}@edu.pucrs.br

Abstract

This paper is related to a practical project of deep learning discipline of Master's in Computer Science of the Pontifical Catholic University of Rio Grande do Sul. We considered that emails are a good source of non-structured data, and that are some automation opportunities. Therefore, we fine-tuned the BERT model to reply emails in the English language. The final model could not be evaluated using a quantitative metric. However, a more subjective metric showed that, in general, the model still presents some problems that allow the human being to recognize when an AI system is replying to an email.

1. Introduction

If we consider the scenario of a sales representative who sells software licenses, for example, it is common to receive requests for quotations from their customers about the price of software licenses. There are some opportunities to provide some automation on this task. Therefore, aiming to automate the answering emails task, we implement in this paper an agent that is able to reply an email using a deep learning model.

We can implement this agent with two primary components: one to read the email messages and send the responses and another to receive the email messages parse it and prepare the response based on message content. In this paper, we deal with the piece of software responsible for receiving the message content and generate the proper reply for that message.

A significant amount of software uses Natural Language Understanding (NLU), and they achieve excellent results in understanding the context of messages and generating a proper response to that. Looking for all different approaches that we can use on natural language understanding, dataset availability, and time to implement, we choose to fine-tune a BERT model for the sequence-to-sequence (Seq2Seq) task.

The first idea was to implement something using the Brazilian Portuguese language, and we have looked for some datasets available. We could not find any suitable source of data to train our model. We know that there are many more datasets in the English language available, so we decide to go that way.

We did not implement the piece of checking email inbox and sending the answer email. We consider this task; it is trivial, and it is not the objective of this assignment.

Our goal for this assignment is to provide software capable of receiving any text with some context and producing a meaningful answer to that. To make it happen, we will use a public dataset of emails with responses to train our model and do some experiments. Later on, we will submit a survey to a group of people asking them to identify based on some questions when the answers were from a human or computer software.

In the Section 2, we present some the background about the BERT model and the architecture used in this paper. Section 5 discusses how we parsed the dataset and the results achieved with the fine-tuned model.

2. Background

For this assignment, we need to understand concepts about language model, sequence-to-sequence (Seq2Seq), attention mechanisms, and transformers. As we saw in classes, when we have a sequence of words, we need to compute the distribution of probabilities of the next word of sequence based on specific vocabulary. This system is known as a language model. Simply we consider language model as a system that defines probabilities to a piece of text. There are some issues with language models based on n-grams: sparse, storage of count of n-grams in memory. Another possibility is to use a trigram approach, but we saw that although the result generated is grammatically complaint on the English language, we do not have a context. So this is not what we need, and we learned too about sequence-to-sequence (Seq2Seq).

BERT

BERT is a short for Bidirectional Encoder Representations from Transformers, proposed by (Devlin et al. 2018). Transformers network was proposed by (Vaswani et al. 2017) and use the attention mechanism, proposed by (Bahdanau, Cho, and Bengio 2015). In a nutshell, the attention mechanism can learn representations between words that express their contextual meaning.

Since BERT is bidirectional, the attention mechanism does not learn only representations of just the previous words, but for all the words in the sentence. To achieve this, BERT was pre-trained for masked language modeling

and next sentence predictions tasks. However, with the adjustment of the output layers and a fine-tuning process, the model can be used for other tasks, using the same parameters as the original BERT model.

The original BERT model was pre-trained in a corpus comprising Wikipedia and Book Corpus. Two pre-trained models are available: BERT large, with 345M parameters and 24 layers, and BERT base, with 110M and 12 layers.

Figure 1 shows the architecture of a Transformer network, proposed by (Vaswani et al. 2017). Originally, Transformers was proposed to be used for machine translation tasks, so the architecture has encoder and decoder blocks. BERT uses this same architecture, but trained with different tasks, as previously stated.

The next two sections discusses the BERT model encoder and decoder blocks, used for sequence-to-sequence tasks.

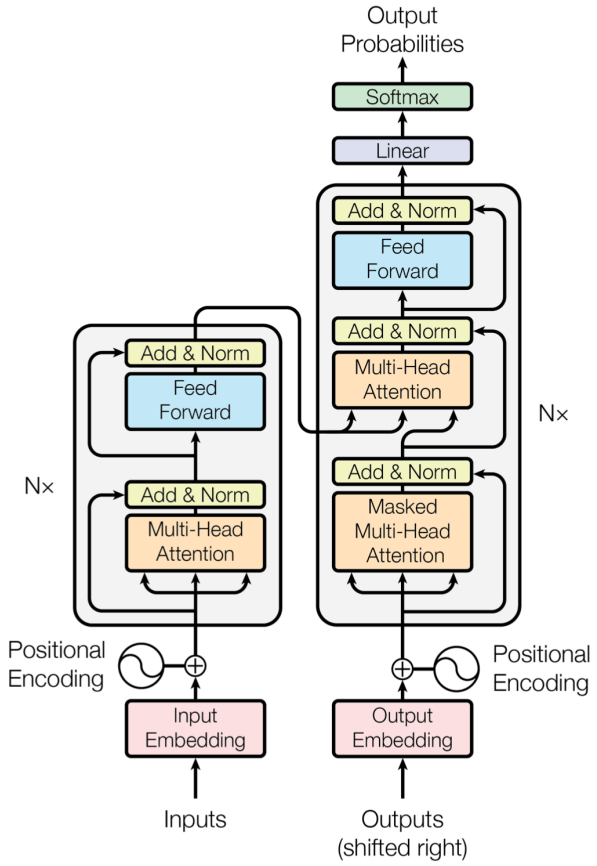

Figure 1: Transformer Architecture

Source: (Vaswani et al. 2017)

BERT encoder

In the first step, each token in the sequence is embedded in a N dimmensions vector. The position of the token in the sentence is also taken into account. Therefore, after embedding the tokens, positional encoding is added to the embedded matrix. In BERT, the embedding and the position encoding is learned through back-propagation.

The next block in Figure 1 is the encodding block. This block can be chained n times to, intuitively, learn more complex relationships between words. The BERT base architecture chains 12 of these blocks.

BERT decoder

The embedding of the the decoder block is similar to the encoder block. The difference is that the target sentence is shifted with a token to indicate the start and the end of the sentence.

The masked self-attention used in the decoder ignores the words on the right. That is, since the task used in the original Transformer is a language model, the words on the right represents the word that the model needs to learn to predict and should be ignored. This is not true for BERT, as it uses different tasks to train the model.

For BERT, the decoder architecture is almost the same as the encoder architecture. The difference is that, for multi-head attention, the output of the encoder network is used.

3. Related work

We know that there is plenty of work done in this area, so we looked at what others already implemented to have insights and technical information on how to achieve our goal. At first glance, we checked some features implemented by Google on their email service and what theoretical foundations they used. On of the existing features available at Gmail is the Smart Reply that it is an automated response suggestion that users can choose just taping on the suggested text. The concept of Smart Reply is to generate a short reply message looking at the context of the email[cite]. This concept is a good start, but we intend to provide a more detailed answer to our emails. [cite] proposes a different approach to answer generation that it is more aligned with what we intend to achieve. Although the method used seems to be more efficient, we decided to pursue a more straightforward implementation due to our constraints of datasets, computational resources, and time.

4. Experiments and Results

In this section, we discuss the how the dataset and the implementation of the sequence-to-sequence model was handled and the results achieved.

Dataset

As the project focus is on automatic email reply, The Enron Email Dataset[1] was used to train the model. The dataset contains only the emails raw data. Therefore, a parser[2] was built to extract the email and the replies from the raw data of the email.

To identify whether an email has a reply or not, we look for emails that contain the string `-----Original`

[1] https://www.kaggle.com/wcukierski/enron-email-dataset

[2] https://www.kaggle.com/claudioscheer/extract-reply-emails

`Message-----`. After filtering only emails with non-empty replies, those emails were parsed into an input sequence (the original email) and a target sequence (the reply email). The entire extraction was done automatically, that is, we did not extract or adjust any email manually.

Two libraries were used to parse the dataset: `talon`[3], provided by Mailgun, and `email`, provided by Python. The `email` package returns the email body with the entire thread. To extract only the last reply from an email thread, the `talon` package was used.

The original dataset contains 517 401 raw emails. After parsing the raw dataset, the new dataset consisted of 110 205 input and target pairs. As the resources available to fine-tune the model were limited, only emails with less than 256 characters were used. The final dataset consisted of 40 062 emails. All of these input and target pairs were used to train the BERT model.

21 emails that were not correctly parsed were used to evaluate the model and obtaing the BLEU score. These emails were chosen manually from the dataset.

Implementation

A pre-trained BERT model, provided by Hugging Face[4], was used. Hugging Face also provides a PyTorch library for using the pre-trained models. Therefore, this library was used to implement the sequence-to-sequence model, with PyTorch 1.5.1.

The `BertModel` class provided by Hugging Face can behave as an encoder or decoder. The difference is that, for the encoder, only a layer of self-attention is used and, for decoder, a layer of cross-attention is added between the layers of self-attention. The difference between these attention mechanisms is discussed in Section BERT.

In this paper, the BERT base architecture was used. This model uses fewer resources, which allowed us to increase the batch size. To fine-tune the model, the following hyperparameters was used:

- Learning rate: $1e{-}4$;
- Warm-up steps: 5000;
- Epochs: 10.5;
- Adam epsilon: $1e{-}4$, same as in the original BERT paper (Devlin et al. 2018);
- Batch size: 10;
- Beam search hypothesis: 3;

In the fine-tune process, an EC2 spot instance on AWS was used. A checkpoint of the model was saved every 20 000 steps, mitigating the loss of processment if the instance was suddenly stopped. The instance was equipped with a NVIDIA T4 GPU (CUDA 10.2), with 16GB of memory.

The batch size was limited by the amount of GPU memory available. We also tested to accumulate and update the gradients after 4 steps, but we did not see any improvements.

[3]https://github.com/mailgun/talon

[4]https://huggingface.co/

Different values were tested for the learning rate and epochs, with and without warm-up. Figure 2 shows the final results of the learning rate update during the fine-tuning process.

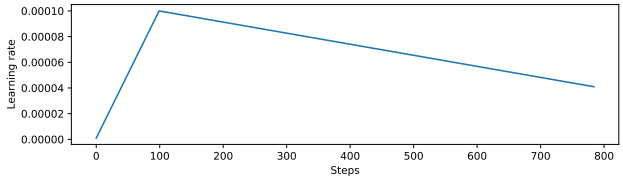

Figure 2: Learning rate schedule

Figure 3 shows the loss function value over the fine-tuning process. We do not know why, but loss function shows a greater decrease at the end of each epoch.

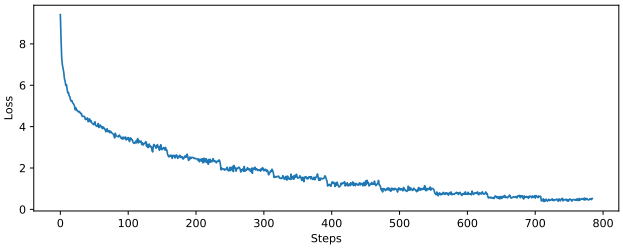

Figure 3: Loss function

Some hyperparameters configuration shows bad results in the loss function. For example:

- learning rates greater than $1e{-}4$ did not make the model converge;
- the use of a warm-up period allowed the model to converge in less epochs;
- a high number of epochs almost causes an overfitting of the model;

The last changed hyperparameter was the number of hypothesis explored in each branch of the beam search. A number greater than 3 resulted in more words being out of context in the generated reply email.

Results

The fine-tuned model still has some noise in the replies generated. Therefore, only the first part of the text of the generated replies was used. This is valid for the BLEU score and for the subjective evaluation.

The BLEU score was used to get a quantitative result of the model. Using the evaluation dataset, the BLEU score was 0.0. This does not mean that the replies were bad. This means that the generated replies do not match to the replies originally sent. Table 2 shows some examples of why the BLEU score was 0.0.

As shown in Table 2, BLEU is not a good option for this problem. Therefore, a subjective assessment of the replies was made. We generated some replies, created a form using

Email	Reply	AI?	Was it AI?
Let me know if this works.	This works great. Will let you know. as soon as I get a chance, let me know	95.7%	yes
Following are the terms under which I ran the Forward Obligation report and received this message: General SQL error. ORA-00907: missing right parenthesis. Terms: Let me know what you find. Thanks!	Kate, I ran the report with the exact same options and it came right up. Try it again, exactly as before. Usually if it works for one person and not the other it is a security issue. Let me know what happens. Thanks, Brettther	52.2%	yes
Didn't you trade uranium at one time?	Yeah, I know the buisness VERY well.	30.4%	no
Are you free for drinks either Monday or Wednesday?	Yes	69.6%	no
Mons, I would be available on the 25th, 26th or 27th. I cannot make it the week of the 18th. Thanks, Bill.	OK, so, let's see if we can get together later today. I have to leave at 16:00 for a few minutes, but I am sure that I will be out at that moment. Thank you Kim.	39.1%	yes

Table 1: Is it AI? test

Input	Target	Generated
nevermind. are you at work?	yea	i'm all alone over here...
what re your plans for tomorrow?	drop prentice's car off at shop; go to dome	leaving early tomorrow. leaving early. leaving early. leaving early.
I just refaxed it. For your information, my total hours is 49.	I noted your hours..I will look for the t/s Thanks Lisa!	Thanks Rick, and let's talk after the holidays about a new game plan... Best regards and happy thanksgiving. Jaime Williams.

Table 2: Example of generated replies

Google Forms, and asked some people to answer the questionnaire.

The question asked for each reply generated was: Was this reply generated using Artificial Intelligence software? The interviewee could choose only yes or no. The form contains only five emails and replies. The Table 1 shows the results of the test answered by 23 human beings. The column 'AI?' shows the percentage of answers that believed the reply was generated by an AI system. The column 'Was it AI?' shows whether the reply was generated by an AI system or not.

In general, Table 1 shows that human beings can identify that the reply was generated by an AI system. Also, it is important to note that when the reply is short, most people think that it is generated by an AI system.

5. Final Remarks and Future Work

Despite the small dataset used and limited resources available, the fine-tuned model performed well. In a further works, the dataset must be revised to avoid data that may cause noise in the predictions.

References

[Bahdanau, Cho, and Bengio 2015] Bahdanau, D.; Cho, K.; and Bengio, Y. 2015. Neural machine translation by jointly learning to align and translate. In Bengio, Y., and LeCun, Y., eds., *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

[Devlin et al. 2018] Devlin, J.; Chang, M.; Lee, K.; and Toutanova, K. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR* abs/1810.04805.

[Vaswani et al. 2017] Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is all you need. *CoRR* abs/1706.03762.