6312 – Natural Language Processing  
Final Project Report  
Salim Haruna

05/02/22

Table of Contents

[1.0 Introduction 2](#_Toc101394297)

[2.0 Dataset Overview 2](#_Toc101394298)

[3.0 Text Preprocessing 2](#_Toc101394299)

[4.0 Transformer Model 3](#_Toc101394300)

[4.1 Encoder 4](#_Toc101394301)

[4.2 Decoder 4](#_Toc101394302)

[4.3 Bert Pretrained Model 5](#_Toc101394303)

[5.0 Experimental Setup 6](#_Toc101394304)

[6.0 Results 7](#_Toc101394305)

[7.0 Conclusion 8](#_Toc101394306)

[8.0 Citations 8](#_Toc101394307)

# 1.0 Introduction

COVID-19 is a serious global infectious disease outbreak with nearly 550,000 cases and around 25,000 deaths worldwide. It is part of a family of viruses called coronaviruses that infect both animals and people. This particular one originated in China at the end of 2019, in the city of Wuhan, which has 11 million residents.

Long before COVID-19 was identified, a lot of people struggle to believe information when it comes to public health due to various reasons. Maybe a good way to handle this is to make more scientific information available to readers. Therefore, I imagined having a question-and-answer application model where anyone can ask questions related to COVID-19 and a simple summarized answer is returned citing scientific research. By adopting this method, it saves people the time of having to look through entire internet websites and streamline it down to covid-19 articles providing summarized answers and pointing to the articles these answers were generated from.

# 2.0 Dataset Overview

The dataset that we will use is sourced via Kaggle and was put together by the white house and some of the leading research groups. The dataset is 16 GB of text data consisting of csv and json files. It is a resource of over 134,000 scholarly articles, including over 60,000 with full text, about COVID-19, SARS-CoV-2, and related coronaviruses. The dataset also has web links to the full text of each article by uses the doi number. One of the most important files on this data is the metadata which has the following columns:

*cord\_uid, sha, source\_x, title, doi, pmcid, pubmed\_id, license, abstract, publish\_time, authors, journal, mag\_id, who\_covidence\_id, arxiv\_id, pdf\_json\_files, pmc\_json\_files, url, s2\_id*

# 3.0 Text Preprocessing

As discussed above the dataset is very large. Training a model on that data will be computationally expensive. A better option was to streamline the dataset by using the meta data and only selecting the most important fields like title, doi, authors, abstract etc.

In order to generate the dataset for model training, I loaded the metadata csv file into a pandas data frame and ran an iterator. On each iteration, I got the doi number and appended it to the doi URL and sent a web request to return the webpage containing the article. I used beautiful soup to extract the text from the webpage and appended it to a new column labelled body text.

The user questions input is in two forms, the text and speech. The text is directly passed to the transformer model without any preprocessing, but the speech is recorded and written to a .wav file and the google translate interpreter is used to convert the .wav file to text and then it is passed to the transformer model.

# 4.0 Transformer Model

The transformer model is a neural network that learns context and meaning by tracking relationships in sequential data such as the words in sentences. These models apply self-attention mathematical techniques, to detect subtle ways even distant data elements in a series influence and depend on each other.

Diagram

Description automatically generated

The Transformer consist of an encoder-decoder structure.The task of the encoder is to map an input sequence to a sequence of continuous representations, which is then fed into a decoder while that of the decoder is to receive the output of the encoder together with the decoder output at the previous time step, to generate an output sequence.

## 4.1 Encoder

The encoder first sublayer implements a multi-head self-attention mechanism which implements ℎ heads that receive a linearly projected version of the queries, keys and values each, to produce ℎ outputs in parallel that are then used to generate a final result and the second sublayer is a fully connected feed-forward network, consisting of two linear transformations with Rectified Linear Unit (ReLU) activation in between.

## 4.2 Decoder

The decoder shares several similarities with the encoder and composed of three sublayers where the first sublayer receives the previous output of the decoder stack, augments it with positional information, and implements multi-head self-attention over it. The decoder is modified to attend only to the preceding words. Hence, the prediction for a word at position i, can only depend on the known outputs for the words that come before it in the sequence. The second layer implements a multi-head self-attention mechanism, which is similar to the one implemented in the first sublayer of the encoder. On the decoder side, this multi-head mechanism receives the queries from the previous decoder sublayer, and the keys and values from the output of the encoder. This allows the decoder to attend to all of the words in the input sequence.Lastly, the third layer implements a fully connected feed-forward network, which is similar to the one implemented in the second sublayer of the encoder.

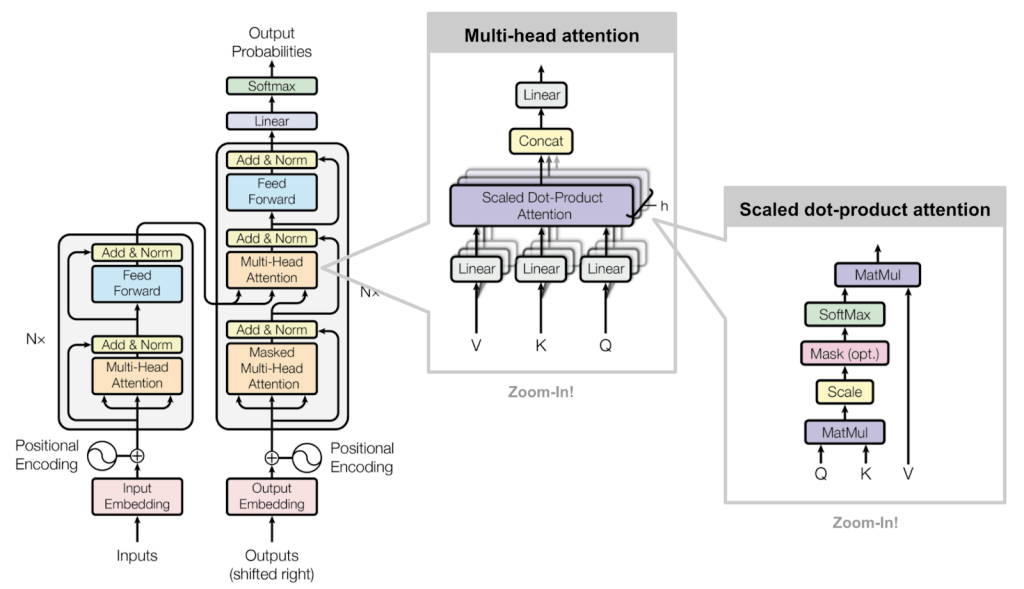
Furthermore, the three sublayers on the decoder side also have residual connections around them, and are succeeded by a normalization layer.Positional encodings are also added to the input embeddings of the decoder, in the same manner as previously explained for the encoder.

In summary the Transformer model runs in the following processes:

1. Each word forming an input sequence is transformed into a n-model-dimensional embedding vector.
2. Each embedding vector representing an input word is augmented by summing it (element-wise) to a positional encoding vector of the same dmodel length, hence introducing positional information into the input.
3. The augmented embedding vectors are fed into the encoder block, consisting of the two sublayers explained above. Since the encoder attends to all words in the input sequence, irrespective if they precede or succeed the word under consideration, then the Transformer encoder is bidirectional.
4. The decoder receives as input its own predicted output word at time-step, t–1
5. The input to the decoder is also augmented by positional encoding, in the same manner as this is done on the encoder side.
6. The augmented decoder input is fed into the three sublayers comprising the decoder block explained above. Masking is applied in the first sublayer, in order to stop the decoder from attending to succeeding words. At the second sublayer, the decoder also receives the output of the encoder, which now allows the decoder to attend to all of the words in the input sequence.
7. The output of the decoder finally passes through a fully connected layer, followed by a softmax layer, to generate a prediction for the next word of the output sequence.

## 4.3 Bert Pretrained Model

BERT (Bidirectional Encoder Representations from Transformers) is a recent paper published by researchers at Google AI Language. It has caused a stir in the Machine Learning community by presenting state-of-the-art results in a wide variety of NLP tasks, including Question Answering (SQuAD v1.1), Natural Language Inference (MNLI), and others.



BERT is a transformers model pretrained on a large corpus of English data in a self-supervised fashion. This means it was pretrained on the raw texts only, with no humans labelling them in any way (which is why it can use lots of publicly available data) with an automatic process to generate inputs and labels from those texts. More precisely, it was pretrained with two objectives:

* Masked language modeling (MLM): taking a sentence, the model randomly masks 15% of the words in the input then run the entire masked sentence through the model and must predict the masked words. This is different from traditional recurrent neural networks (RNNs) that usually see the words one after the other, or from autoregressive models like GPT which internally mask the future tokens. It allows the model to learn a bidirectional representation of the sentence.
* Next sentence prediction (NSP): the models concatenate two masked sentences as inputs during pretraining. Sometimes they correspond to sentences that were next to each other in the original text, sometimes not. The model then has to predict if the two sentences were following each other or not.

BERT makes use of Transformer, an attention mechanism that learns contextual relations between words (or sub-words) in a text. In its vanilla form, Transformer includes two separate mechanisms — an encoder that reads the text input and a decoder that produces a prediction for the task. Since BERT’s goal is to generate a language model, only the encoder mechanism is necessary.

As opposed to directional models, which read the text input sequentially (left-to-right or right-to-left), the Transformer encoder reads the entire sequence of words at once. Therefore it is considered bidirectional, though it would be more accurate to say that it’s non-directional. This characteristic allows the model to learn the context of a word based on all of its surroundings (left and right of the word).

# 5.0 Experimental Setup

To start training using the transformer models, I loaded the following into the main class.

1. From the data preprocessing, the generate dataset is loaded.
2. The bert-base-cased tokenizer is loaded
3. The facebook/bart-large-cnn summarizer is loaded
4. The bert-base-cased huggingface transformer model.

After loading the models, the articles are vectorized using the TF-IDF vectorizer while using the cosine similarity to read through the title, abstract and text of the articles and get the top n most articles related to that question based on the similarity score.

After retrieving the articles, then a batch size is created based on the length of the body text. A loop is run on the batch size and the question and the body text for the row in the iterator is passed to the transformer model. The question and body text and batch encoded with special tokens and padding, and tensor inputs are generated for the model. The input is passed into the transformer model. The model returns start logits and end logits which are used to determine the most likely beginning and ending of the answers are also generated based on the argmax score.

The answers are returned based on the start score and the answer text is now retrieved using the model decoder. For each of the generated answer, the text is passed to the summarizer model and limited to only 100 words to make it as simple as possible for users to read and understand.

# 6.0 Results

In order to arrive at the final results, we achieved several results from data preprocessing to model training.

During data preprocessing, when a user selects the speech input, the recorder starts and uses the system default microphone to record the question. The question is saved as a .wav file. During translation to text, the .wav file is broken down to chunks for every 60 seconds.

During program execution, the bert-cased transformer model is loaded and written to the output folder if the model does not exist in the directory. The files written are config.json and the model.bin.

When the program performs TF-IDF cosine similarity and the model analysis the question, a dataframe of answers ranked by start score is generated and exported as anwers.csv.

For example, the question “Are there any drugs for Covid-19?” and below is a answers dataframe.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Title** | **Source** | **Authors** | **URL** | **Publish Time** | **Doi** | **Start Score** | **End Score** | **Summarized Answer** |
| Repurposing Therapeutics for COVID-19: Rapid Prediction of Commercially available drugs through Machine Learning and Docking | medrxiv | Sovesh Mahapatra; Prathul Nath; Manisha Chatterjee; Neeladrisingha Das; Deepjyoti Kalita; Partha Roy; Soumitra Satapathi | <https://doi.org/10.1101/2020.04.05.20054254> | 2020-04-07 | 10.1101/2020.04.05.20054254 | 303 | 349 | Coronaviruses are classified into four classes designated as alpha, beta, gamma, and delta. 14 SARS - CoV - 2 spikes also bind to receptors on the human cell surface called angiotensin. |
| Chloroquine and hydroxychloroquine in the treatment of COVID-19 with or without diabetes: A systematic search and a narrative review with a special reference to India and other developing countries | Elsevier | Singh, Awadhesh Kumar; Singh, Akriti; Shaikh, Altamash; Singh, Ritu; Misra, Anoop | <https://doi.org/10.1016/j.dsx.2020.03.011> | 2020-06-30 | 10.1016/j.dsx.2020.03.011 | 98 | 355 | People with diabetes and COVID - 19 may need special attention and clinical care. Reports gathered so far have suggested that a number of drugs could be potential candidates for the treatment. The clinical effectiveness of these drugs have not yet been fully evaluated. HCQ has been approved in the treatment of type 2 diabetes in India since 2014. |

After the model generates the answer, a summarizer is used to simplify the answer.

# 7.0 Conclusion

The use of transformer models has provided very great results efficiently. One benefit of using the transformer model is due to the multi-head attention mechanism that gives the network the ability to pass through multiple words simultaneously. The reason I selected bert is because it can be pre-trained on a massive corpus of unlabeled data, and then fine-tuned to a task for which you have a limited amount of data. This allows BERT to provide significantly higher performance than models that are only able to leverage a small task-specific dataset.

# 8.0 Citations

1. <https://huggingface.co/bert-base-cased>
2. <https://huggingface.co/docs/transformers/tasks/question_answering>
3. <https://huggingface.co/docs/transformers/tasks/summarization>
4. <https://www.gavi.org/vaccineswork/what-is-covid-19-and-how-does-it-spread?gclid=CjwKCAjw9e6SBhB2EiwA5myr9vipbbHCxlKWGmVhvNYuWIPNexzxpJGhBC2WGjOOhnwaWcVAaGiJjhoCUoMQAvD_BwE>
5. <https://blogs.nvidia.com/blog/2022/03/25/what-is-a-transformermodel/>
6. <https://machinelearningmastery.com/the-transformer-model/>