AI6127 Group Project Report

Leveraging Large LM for Question and Answer Chatbots

Team Members:

- Lim Jia Xiang (Matric No: G2102097K) (email: s210033@e.ntu.edu.sg)
- Ng Ngee Khung (Matric No: G2101970F) (email: s210026@e.ntu.edu.sg)
- Qi Meng (Matric No: G2204604G) (email: mqi001@e.ntu.edu.sg)
- Sean Goh Ann Ray (Matric No: G2202190G) (email: sean0057@e.ntu.edu.sg)

1. Abstract

In most chatbot applications that do question and answering, the replies tend to be in a very structured format rendering the chatbot less “human-like”. In this project, we explore ways to create a chatbot that is able to generate replies in a manner that resembles how a human reply would be like. In our work, we created a chatbot that can distinguish between a yes/no and open-ended type question and generate a more human-like response for open-ended type of question. We fine-tuned a pre-trained GPT2 model on the specific question answering dataset from amazon. We created our own pipeline and architecture that enables the application to fine-tune to any specific dataset for questions and answers, potentially allowing multiple smaller enterprises to fine tune a large language model to a specific domain within the enterprise use case.

2. Introduction

With the rise in internet businesses and services, more websites are utilising readily-available chatbots to assist customers and users with their online experience. Unlike a frequently-asked-questions page, users may have queries specific to their situation, and going through a customer-service hotline may take some time, assuming it is during business hours. By using chatbots, users may get a response to their query immediately and available 24 hours 7 days. Chatbots also allow multiple users to get their queries answered at the same time, unlike a customer-service hotline where there is a limit to the number of assistances.

However, most chatbots are designed rigidly. They might not be able to understand the human query, especially if it is too specific, too vague, or contains unseen words. These chatbots will compare the query to the database that it was trained on, get the most similar question and take the answer of that question as the answer to the query, which may not be the answer that the human was looking for. This problem grows with the number of users and variety of queries, especially if the chatbot is not updated to the new queries.

With advances in Natural Language Processing (NLP) and large language models, chatbots are becoming more human-like and able to provide more accurate answers. To leverage on this, we introduce an architecture that consists of 3 language models, where each model has a defined task to accomplish. Any query a user has would be input into the first model, which classifies the query into either an open-ended question or a yes/no question. Depending on the classification, the query is then sent to the other models, another classification model if it’s

a yes/no question, and a GPT model if it’s an open-ended question. By using pre-trained models and fine-tuning them on a specific dataset, we hope to achieve a comparable accuracy whilst having a more human-like response.

3. Related Works

Chatbots typically use machine learning algorithms and language models to understand and generate conversations. Among them, machine learning algorithms can learn from training data to improve the conversational ability of chatbots. Language models, on the other hand, can help chatbots understand the meaning of human language so that they can more accurately respond to human questions or commands.

There are a many chatbots that have gained widespread adoption: such as Siri developed by Apple, Google Assistant developed by Google, Alexa developed by Amazon and Cortana developed by Microsoft, etc. Different chatbots may use different algorithms and models and may be customized for specific use cases. For example, Siri's main function is to help users complete various tasks, such as sending text messages and finding information, while Google Assistant is mainly used to answer users' questions.

Chatbots usually consist of the following parts:

- Voice/text input: This is the way users send messages to the chatbot. Voice input requires the use of speech recognition technology to convert the user's speech into text. Text input, on the other hand, receives text input directly from the user.
- Natural Language Understanding (NLU): This is the process of converting the natural language input by the user into a machine understandable language representation. This process requires the use of NLP techniques, such as word segmentation, part-of-speech tagging, named entity recognition, sentiment analysis, etc.
- Knowledge base/corpus: This is the chatbot's knowledge source, including various predefined questions and answers, corpora, databases, and more. The chatbot needs to get answers and reply to the user by querying these resources.
- Reply generation: This is the process of translating the machine-understood intent into natural language. This part usually uses natural language generation (NLG) techniques, such as text generation, template matching, rule matching and other methods, to generate the machine's reply.
- Dialogue management: This is the part that manages the conversation between the chatbot and the user and is responsible for combining the user's input with the machine's reply into a dialogue flow. Dialogue management is often implemented using state-machine based or deep reinforcement learning based algorithms.

Chatbots are complex systems that need to cover the complete flow from user input to machine reply, while multiple techniques and algorithms need to be used to support these functions.

4. Approach

We handle the question and answering problem in a multi-stage process where the application will first take in an input from a user. Some basic text cleaning would be done to remove special characters or symbols before passing the cleaned text to the next stage.

In the next stage, a BERT classification model will be utilised to classify the question into either yes/or or open-ended type. Depending on which class the question is classified into, the application will route the question to a different path. If the question is classified as a yes/no type, then it will be routed to another BERT classifier which shares the same architecture as the previous BERT classifier with the previous one with only the classification layer differing from each other.

The BERT encoder is shared between the two classifiers in order to reduce memory and computational complexity. If the question is of open-ended type, the application will route to a fine-tuned GPT2 model to perform an auto regressive task where the answer will be generated based on the question input to the model. Figure 1 illustrates a high level overview of how the application works.

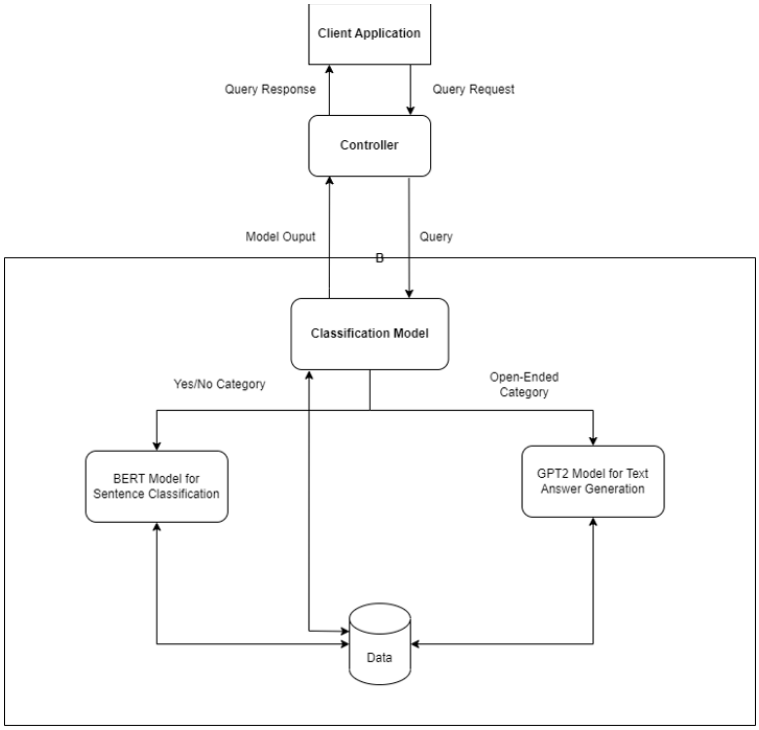

Figure 1: Proposed Architecture

In this application, the user will send a question to the classification model, and it will be classified into a yes/no or open-ended question type. This classification model is made up of pre-trained distilBERT model which is a Bidirectional Encoder Representations from Transformers (distilBERT-base-cased model). The question will be encoded into a dense vector and sent to a classification layer to be classified (with a sigmoid function after the feed forward neural network) into whether it is yes/no or open-ended question type. Figure 2 illustrates the architecture of the classification model.

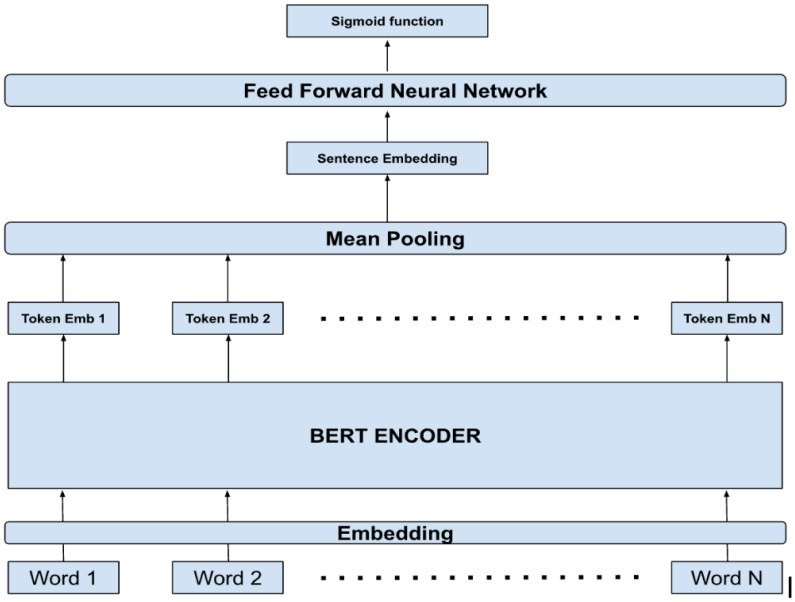

Figure 2: Classification Models Architecture

In this portion, we keep the pre-trained distilBERT model as it is without any fine-tuning done to the weights and simply use the output of the sentence encoding created by the distilBERT model. As the distilBERT model would output an embedding for each word token, in order to create a single embedding representation of the sentence, all the token embeddings are mean pooled. The classification layer is then trained on the task of classifying the question type.

After the first model has classified the question into its respective question type, depending on whether the question is a yes/no or open-ended type, it shall be routed to another binary classification model or a text generation model. If the question is classified as a yes/no question type, the sentence embedding from the previous classification model will be reused (to reduce memory and computational complexity). A different classification layer will be used to classify the question into yes or no which will be the answer received by the end user.

If the question is classified as an open-ended type, it will then be directed to a fine-tuned GPT2 model which will generate text and return to the end user as the answer to their question. We utilised a pre-trained model “gpt2” from hugging face and fine tuned it to our question and answering dataset. During the training process, for each question and answer set, we used another pre-trained distilBERT model to retrieve top 5 question and answer data that is most contextually similar to it.

We utilised FAISS, a library developed by Facebook for efficient embedding retrieval. We then concatenate the 5 questions and answer to the original question and answer pair to form an “augmented” data to fine tune our model with. The reason for doing this is to add more contextual information to the training data and also the dataset is formulated in such a way that there could be multiple answer to the same set of question, hence by concatenating the top few most similar question and answer pair, it could theoretically improve the model performance during inference time. Figure 3 below illustrates the architecture of the GPT2 model.

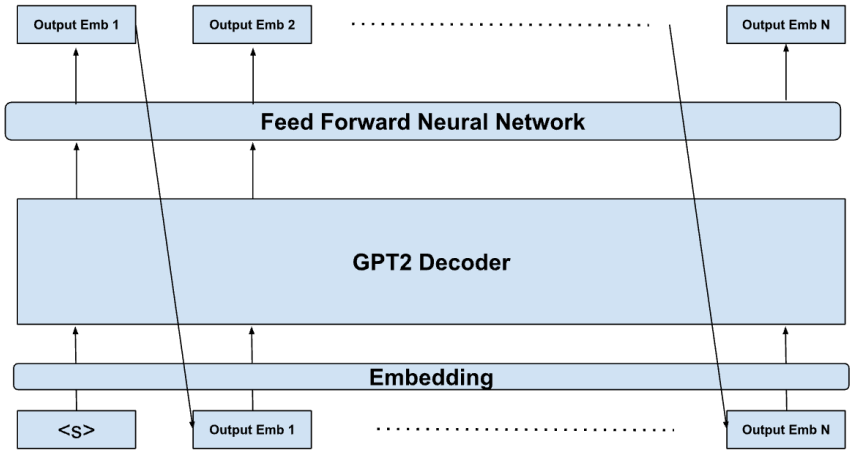

Figure 3: Open-ended Model Architecture

We can observe from the figure that GPT2 is an autoregressive model where the output from the previous time step will be the input for the current timestep. This has an effect on the computation time, especially if the model was loaded only to the CPU.

5. Experiments

The project codes can be found in the following repository: [Github URL]

5.1 Data

The dataset used in the experiment is the Amazon Question/Answer Dataset, which contains approximately 1.4 million answered questions over 17 different categories. This data was collected and made available by Prof. Julian McAuley of University of San. Diego (UCSD). Some statistics of the dataset is shown below in Table I.

Questions:	1.4 million
Answers:	4,019,744
Labeled yes/no questions:	309,419
Number of unique products with questions:	191,185

Table I: Amazon Question/Answer Dataset Statistics

However due to limit in computation resources, a total of 23,000 question and answer of both question text were sampled to train our models.

5.2 GPT2

The expected input during fine-tuning of the GPT2 model is in the following format:

```
QUESTION: question 1 ANSWER: answer 1 ..... QUESTION6 : question 6 <|sep|>
ANSWER: answer 6.
```

Question and answer pair number 6 is the original question and answer data while the question-and-answer pair one to five are pairs that are semantically like the original question. The most original question is position at the end of sentence in order to teach the GPT2 model

during fine-tuning that it is the most crucial portion in the input data. Each data pair will have a prefix of either QUESTION or ANSWER prepended to it such that the model is able to understand which portion of the text represent the question and answer. The |<sep>| token separate the text data into context and answer.

During Inference time the data into is of the following format:

```
QUESTION: question  <|sep|>
Given the data in this format the fine-tuned GPT2 should out put the answer in
the following format:
ANSWER: answer
```

5.3 Evaluation Method

We use the following popular n-gram based metrices to evaluate the open-end question-and-answering:

- BLEU is used to evaluate machine translation as a precision-based metric. It scores an output by computing the number of n-grams in the output that also appearing in the reference text. The 'n' can vary from 1 up to the specified N value. The different scores for the varying n are then aggregated with a geometric mean.
- ROUGE is being used to evaluate machine translation and summarization. In this project, we focus on ROUGE-1 which measures the matching rate of unigrams between the generated text and the reference text.

5.4 Model Settings

Table II shows the model configurations that are used for the Question Type classification, Yes/No classification and the open-ended Q&A:

	Question Type Classification	Yes/No Classification	Open-ended Q&A
Pre-trained Model	DistilBert-base-cased	DistilBert-base-cased	GPT2
Learning Rate	1e-6	1e-6	2e-5
Training time per epoch	1~2 minutes	1~2 minutes	33 minutes
Number of epochs trained	20	20	7
Batch Size	32	32	4
Number of question answer pair augmentation (most similar Q&A pair)	NA	NA	5

Table II: Model Settings for the Different Models

5.5 Results

The first stage of the architecture consists of a pre-trained distilBERT model, using the distilBert-base-cased embedding weights, to classify the question into either a yes/no or open-ended question type. This model is fine-tuned on the Amazon questions dataset, which was split into 70/15/15 train/val/test sets, and the training loss and accuracy is as shown in Figure 4.

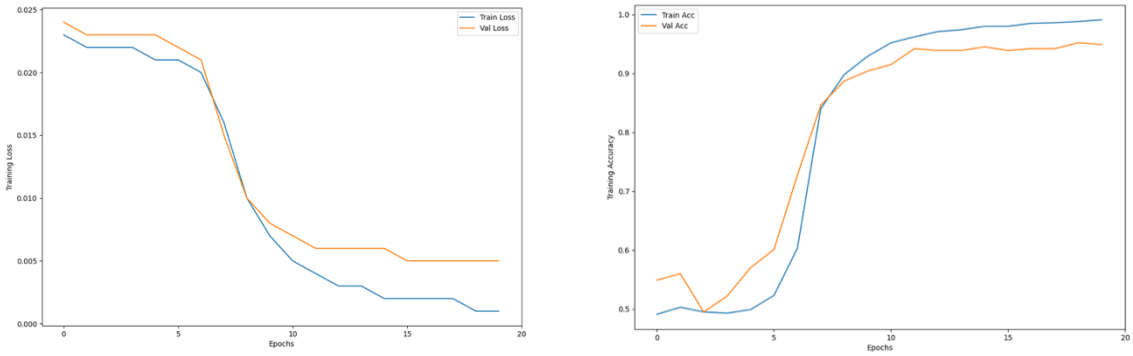

Figure 4: Training Loss & Accuracy on QuestionType Classifier

The fine-tuned question type classifier model achieved 97.3% accuracy on the test set, meaning that it can classify the question type very accurately. The output of the model includes the question ID and their respective question type, which will then be sent to the other models in the proposed architecture.

A second DistilBert model is used to classify yes or no questions. The question of yes/no type distinguished by the first DistilBert model is preprocessed first: By filtering the question ID and the score of the answer, we ensured that each question contained only one answer with the highest score; The text of each answer is divided into a single word by tokenizer. By some simple discrimination methods, such as whether the answer contains yes or no, not, we distinguish all the answers into Y and N two types. The preprocessed data contains 977 different questions, each corresponding to the answer with the highest score, which has been labeled as Y and N, and the second DistilBert model will be trained on this data.

Table III below shows the final performance of the second DistilBert model on the training set, validation set, and test set. We can find that the DistilBert model still performs well when classifying yes or no questions, and after 20 iterations, the model reaches an accuracy of more than 97% on both the training set and the test set. However, it is worth noting that most of the questions are classified as Y, which may be due to underfitting with too little training data, trying to increase the amount of training data can help solve this problem.

	Number of data	Loss	Accuracy
Train set	679	0.003	0.976
Valid set	145	0.006	0.966
Test set	146	0.004	0.973

Table III: Training Loss and Accuracy for the Yes/No Classification

Table IV below illustrates the metrics for GPT2 pre-trained and fine-tuned on the open-ended question answering task. The difference between GPT2 fine-tuned config 1 and 2 is that config 1 uses 5 most similar question and answer pairs whereas config 2 uses only 2.

Model	GPT2 (Base)	GPT2 (Fine-Tuned) Config 1	GPT2 (Fine-Tuned) Config 2
Training Loss	4.23	1.14	2.504
Validation Loss	4.63	4.81	4.268

Training BLEU	0.143	0.242	0.291
Validation BLEU	0	0.023	0.05
Training Rouge F Measure Unigram	0.0101	0.1201	0.1198
Training Rouge Precision Unigram	0.0107	0.1147	0.1183
Training Rouge Recall Unigram	0.0178	0.2247	0.2162
Validation Rouge F Measure Unigram	0.0078	0.1097	0.1065
Validation Rouge Precision Unigram	0.0064	0.1041	0.1050
Validation Rouge Recall Unigram	0.0122	0.2499	0.2291

Table IV: The metrics for GPT2 pre-trained and fine-tuned on open-ended Q&A task

6. Analysis

As we can see, both fine-tuned models outperform the pre-trained model in most of the metrics. However, the score for fine-tuned model is still relatively low with the best validation BLEU at 0.05 only and the validation rouge F measure at 0.1097. The reason for this could be the lack of context in the question-and-answer pair. As the data were sourced from online community, the answers were not curated and could contain a lot of noise, to add on further there was no product details for each question-and-answer pair, making it hard for the model to understand the context of the question.

Also, due to computation limitations, we sampled a total of 26,000 questions and answer pairs only and a large model like GPT2 would require more data to avoid overfitting, which is what is happening in our model training. We can see there is a large gap in some of the training and validation metrics.

Some basic suggestion for overcoming this would be to use more data, which would require more computational resources. Prepending the product along with its description at the start of question could also improve model performance. Lastly certain techniques such as gradient accumulation, higher dropout rates and learning rate warm up or plateau could help with the overfitting issue. We can also adopt some more advanced techniques such as re initialisation of the top few layers of GPT2 and freezing the first few layers of GPT2. Research had shown that the lower layers tend to learn more general features whereas the upper layers specialize more to the fine-tuned task and such re initialisation could improve the model performance on our question-and-answering task in this specific domain.

7. Conclusion

In this project, we explored creating a chatbot that is able to generate replies in a manner that resembles how a human reply would be like. We created a chatbot that can distinguish between a yes/no and open-ended type question and generate a more human-like response for the open-ended type of question. The question type classifier model achieved 97.3% accuracy on the test set, meaning that it can classify the question type very accurately. The DistilBert model performed well when classifying yes or no questions, and after 20 iterations, the model reached an accuracy of more than 97% on both the training set and the test set. And for the open-ended Q&A, the score for the fine-tuned GPT2 model is relatively low with the best validation BLEU at 0.05 only and the validation Rouge F measure at 0.1097. We explained that this could be due to amount of data available and suggested some techniques to improve it.

References

1) Amazon Question/Answer Dataset - https://www.kaggle.com/datasets/praneshmukhopadhyay/amazon-questionanswer-dataset
2) Hugging Face Evaluate Metric - https://huggingface.co/spaces/evaluate-metric/
3) Siri - https://www.apple.com/siri/
4) Personal Digital Assistant - Cortana Home Assistant – Microsoft - https://www.microsoft.com/en-us/cortana
5) What exactly is Alexa? Where does she come from? And how does she work? - https://www.digitaltrends.com/home/what-is-amazons-alexa-and-what-can-it-do/
6) Google Assistant, your own personal Google - https://assistant.google.com/
7) Adamopoulou E, Moussiades L. An overview of chatbot technology[C] /Artificial Intelligence Applications and Innovations: 16th IFIP WG 12.5 International Conference, AIAI 2020, Neos Marmaras, Greece, June 5–7, 2020, Proceedings, Part II 16. Springer International Publishing, 2020: 373-383.
8) Zhang, T., Wu, F., Katiyar, A., Weinberger, K. Q., & Artzi, Y. (2020). Revisiting Few-sample BERT Fine-tuning. *Computing Research Repository (CoRR)*. https://doi.org/10.48550/arXiv.2006.05987

Team Contribution

Name	Description	Percentage
Lim Jia Xiang	Pre-process data for GPT2 fine tuning and fine-tuned GPT2 model for open-ended question & answering type	25%
Ng Ngee Khung	dataset, evaluation metrices	25%
Qi Meng	distilBert Yes/No Classification model	25%
Sean Goh Ann Ray	distilBert QuestionType Classification model	25%