# Answering COVID-19 Questions with a Medical Chatbot Application

**Kate Avery** kavery@cs.umass.edu
**Teodoro Meyerhoff** tmeyerhoff@umass.edu
**Juhyeon Lee** juhyeonlee@cs.umass.edu
**Minhwa Lee** minhwalee@cs.umass.edu

## 1 Problem statement

Delivering accurate medical information to patients is critical for improving their disease management knowledge and facilitating communication between clinicians and patients. However, access to medical information from in-person appointments is limited because of constraints on clinicians' resources and time. Frequent hospital visits are also time-consuming and inefficient, especially for those living in rural areas and underserved communities. Moreover, events like the COVID-19 pandemic can exacerbate the lack of available clinical resources and difficulty of in-person communication between clinicians and patients. Thus, a variety of circumstances can make it difficult for patients to obtain sufficient medical information. To enable frequent and efficient access to medical information via remote healthcare, medical question-answering chatbot systems are potentially valuable future tools for both clinicians and patients. Medical chatbot systems do not require the presence of clinicians and can provide instant answers, regardless of time and place. Moreover, a chatbot can answer simple, repetitive medical questions, provide insights to more open ended queries and identify more urgent and or critical medical questions which they can then automatically forward clinicians, which allows to reserve clinical resources.

This paper aims to build a prototype of a medical question-answering chatbot. Open-domain question-answering on real-world conversational data is a difficult problem, especially for specialized domains, like medicine. Because of the scope of this project, we implemented only a few key pieces of the chatbot. Namely, we fine-tuned a large Transformer-based model on a real-world medical dialogue dataset and attempted to optimize the model. Specifically, we investigated several different architectures described in Section 6.1, and we added a simple information retrieval component described in Section 6.2. The information retrieval component finds the most similar conversation in the training set to the question asked of the model and prepends the conversation to the question. We also applied various preprocessing techniques to improve the model's performance. In the end, we compared our models to a baseline described in Section 5 and evaluated all models using manual human evaluation and automatic measures.

## 2 What you proposed vs. what you accomplished

In general, we accomplished most tasks that we set out to do. We proposed to preprocess the COVID-19 dialogue dataset, build a baseline as described in Section 5, build a model fine-tuned on DialoGPT, and build a "similarity model" as described in Section 6.2.

To make our models perform better, we did some more work than we originally set out to do. We experimented with multiple model types, as described in Section 6.1. We also augmented the COVID-19 Dialogue dataset with data from the more general Medical Dialogue dataset.

We also had a few failures and partial failures. The baseline model copies superficially similar answers from the training set using a simple k-nearest neighbor algorithm, so although the response may not answer the patient's question, it is at least mostly factually correct and relevant to respiratory disease. Our models typically give answers that are related to the question, but they sometimes generate 'patient-like' text or answers with factual errors, which causes them to perform worse in human evaluation. (See Section 7.2 for an in-depth analysis of our models.) Furthermore,

generative models tend to give less fluent answers answers pulled from the training set.

We suspect we experience these issues for two main reasons. First, we fine-tuned a small, less sophisticated model (DialoGPT-small) that would fit in Colab's memory. This small model performs much worse than the medium or large DialoGPT models out of the box. The small version is much more likely to get caught in loops of repetition or to generate nonsensical, out of place text. These shortcomings are evident after a few lines of dialogue. Second, we did not implement a complex information retrieval system, which would likely be necessary to achieve a high level of accuracy. Generative models alone are not very good at open-domain question-answering.

# 3 Related work

## 3.1 Medical Domain-specific Transformer Models

Recently, large, pretrained Transformer models have made advances on various natural language processing (NLP) tasks, including tasks in medical NLP. BERT is one of the Transformer-based models showing state-of-the-art performance on many NLP tasks, such as question-answering (Devlin et al., 2019). However, BERT was pretrained with texts from the general domain and did not perform well on domain-specific texts. The biomedical domain requires much domain-specific knowledge and has a large amount of jargon, which is uncommon in general texts. Thus, several prior studied fine-tuned BERT on biomedical texts, creating models such as BioBERT (Lee et al., 2019), SciBERT (Beltagy et al., 2019), ClinicalBERT (Alsentzer et al., 2019), and BlueBERT (Peng et al., 2019). BioBERT is one of the medical domain-specific BERT language models trained on large-scale biomedical corpora, such as PubMed abstracts and PubMed Central full-text articles. The model has been used as a pre-trained model for transfer learning on various NLP tasks in the biomedical domain, such as biomedical named entity recognition, relation extraction, and medical question-answering (Lee et al., 2019). In this project, we used BioBERT to generate medical domain-specific sentence embeddings for question-answering (QA) task, as described in Section 6.2. We used these embeddings to find semantically similar QA pair to the input question.

## 3.2 Generative Medical Dialogue Models

Generative Transformer-based models have been used for question-answering tasks. For instance, Radford et al. (2019) created the GPT-2 model, which is a Transformer architectures that can generate text, including answers to input questions. Zhang et al. (2020) built the Dialogue Generative Pre-trained Transformer (DialoGPT), which is a variant of GPT-2 fine-tuned on English Reddit conversations (Zhang et al., 2020). Its objective is to maximize the probability of a ground-truth response $X = x_1, \ldots, x_n$ given a dialogue history $S$: $p(X|S) = p(x_1|S) \prod_{i=2}^{n} p(x_i|S, x_1, \ldots, x_{i-1})$. It performs well on back-and-forth naturalistic conversations.

These generative approaches are not frequently used in the medical domain because of their instability and variability, which cannot be tolerated in medical applications (Safi et al., 2020). However, several preliminary works reported using generative models in medical dialogue generation. Zeng et al. (2020) built a medical dialogue model by fine-tuning a pre-trained GPT-2 model (Radford et al., 2019) and a BERT-GPT model Zeng et al. (2020) on a large-scale medical dialogue dataset. Oniani and Wang (2020) fine-tuned a GPT-2 model on a COVID-19 corpus and filtered the answers generated by GPT-2 using the Universal Sentence Encoder (Cer et al., 2018), the Term Frequency-Inverse Document Frequency method (Salton et al., 1975), BERT, and BioBERT (Oniani and Wang, 2020). In this project, we used DialoGPT to build a conversational dialogue system for COVID-19-relevant questions with more human-like answers. Since DialoGPT is pre-trained on natural language data, it produces more natural-sounding dialogue between the user and the system than GPT-2 produces. Further, we experiment with a retrieval component described in Section 6.2.

# 4 Your dataset

## 4.1 Description of Dataset

The CovidDialog-English dataset (Ju et al., 2020) and MedDialog dataset (Chen et al., 2020) were used to train our model. An example data point can be found in Table 1. The CovidDialog-English dataset contains 572 conversations between patients and doctors about COVID-19. The dataset was collected from text chat conversations between patients and doctors on www.icliniq.com

| |
|---|
| **Patient:**<br>Can taking extra vitamin C every day help prevent you from catching the coronavirus or the flu?<br>**Doctor:**<br>In brief: Vitamin C<br>While getting enough vitamin C in your diet is highly recommended, it has no special ability to prevent coronavirus or flu virus.<br>**Patient:**<br>What about extra self care steps, like sleeping more?<br>**Doctor:**<br>... [conversation continues] |

Table 1: An example from the COVID Dialogue dataset, edited for lightly length. Patient questions and doctor responses were often paragraph-length.

and www.healthtap.com. We focus on COVID-19 data because it is abundant and disease-specific. Because of the dataset's specificity, the model does not have to have as much medical knowledge.

We found that our model benefited from having more data, so we augmented the CovidDialog dataset with examples from the MedDialog dataset. The MedDialog dataset contains 229,674 conversations, but we selected only 569 examples related to respiratory diseases like influenza. The MedDialog dataset is structured in the same way as the CovidDialog dataset, so merging the two was not difficult.

For each dataset, the patient spoke for about three to four sentences and the doctor spoke for about five to six sentences per conversation. Only one example in the final dataset had more than four conversation turns (i.e., the doctor responded more than twice).

We were interested in these two datasets because they contain realistic conversations between patients and doctors, which might allow the model to generate more naturalistic answers to questions. Exposing the model to a more difficult dataset is also important if it is going to perform well on the real-world test data. Patients often cannot eloquently articulate their questions, so it is useful for the chatbot to be exposed to realistic and unstructured data during training.

### 4.2 Data Preprocessing

We used the pandas library to do the following preprocessing on our CovidDialog dataset and our MedDialog dataset. Both datasets are comprised of patient-doctor conversations of the form $p_i, d_i$ where i varies. In the example in Table 1, $p_0$ = "Can taking extra...", $d_0$ = "In brief: Vitamin C ...", $p_1$ = "What about extra ...", etc.

#### 4.2.1 Version 1a

In our preliminary experiments, we created a new example for each $d_i$ in the dataset. For example, a conversation with two patient questions and two doctor responses would be split into two examples:
1. Prefix: $p_0$ Ground Truth: $d_0$
2. Prefix: $p_0 d_0 p_1$ Ground Truth: $d_1$

The previous parts of the conversation are often crucial to the future responses, so prepending $p_0 d_0$ is important. The examples from the same conversation were given the same ID number and put in the same train / test / validate split. This preprocessing step increased the size of the dataset to 1141 examples total.

#### 4.2.2 Version 1b

The primary issue with this preprocessing setup was that the chatbot generated "patient-like" text. That is, it asked more COVID-related questions, rather than answering the question. An early attempt to fix this issue involved grouping all the patient text and all the doctor text (e.g. Prefix: $p_0 p_1 d_0$ Ground Truth: $d_1$). This mitigated the issue, but it caused some of the conversations to be out of order.

#### 4.2.3 Version 2

We then tried appending an <eos> token to each each input sequence in Version 1a such that the prefix was of the form $p_i$ <eos> $...d_{i-1}$ <eos> $p_i$ <eos>. This solution showed some promise; the bot now started its responses with doctor-like text. However, these responses were either not very helpful or veered off topic. The bot also seemed to get confused about its identity. For example, the bot might say, "Hello and Welcome to 'Ask A Doctor' service. I have reviewed your query and here is my advice. Yes, you are right. I should definitely consult a doctor. Thanks."

#### 4.2.4 Version 3

Finally, we created special <patient> and <doctor> tokens and prepended them to each input sequence. We also prepended a <doctor> token to the end of the full input sequence, hoping that it would act as a prompt to generate doctor text. An example of the final preprocessing looks like this:
1. Prefix: <patient> $p_0$ <doctor>
Ground Truth: $d_0$
2. Prefix: <patient> $p_0$ <doctor> $d_0$ <patient> $p_1$ <doctor>
Ground Truth: $d_1$

Version 3 was used for our final models. Using this preprocessing, the models produced primarily doctor-like text, though it occasionally generated patient-like text. The final preprocessed dataset was split into 855 training, 142 validation, and 144 test examples (roughly a 75%, 12.5%, 12.5% split). The same split was used for all models. We did not look at the test examples until the final evaluation.

## 5 Simple Baseline

Given a prefix in our test set, we apply a nearest neighbor search algorithm to the space of all sentence embeddings of our training set (padded or truncated to live in $R^{32}$). The baseline returns the answer in our training dataset with the most similar prefix embedding. This algorithm was simple to implement, and the answers returned were only superfically similar to the question asked. For consistency, we used the DialoGPT-small tokenizer to get the sentence embeddings, though another embedding method could be used. The model is located in simple_baseline.ipynb on Github.[1]

## 6 Your approach

Our approach involves training several different models fine-tuned on DialoGPT using a GPU in Colab. First, we experimented with several core models, including an encoder-decoder model, a language model with a causal mask, and a language model with a causal prefix mask (Raffel et al., 2019). The encoder-decoder model was eliminated because it had trouble fitting into Colab's memory. Then, we created a similarity model that prepends the most semantically similar question to the input of the two remaining core models.

We built the foundation of our project by referring to a guide fine-tunes a GPT2 model.[2]. This implementation was a helpful starting point. We used PyTorch to implement the data loading and training loop logic, and then we used HuggingFace functions, like model.train(), model.eval() and loss.backwards(). Once we were able to build out the preprocessing-> training->evaluation->-model-saving pipeline, it was easier to make small adjustments to the underlying architecture or add in new components as the project grew in complexity and scope.

DialoGPT is a GPT2 model fine-tuned by Microsoft on conversational data from Reddit and available through HuggingFace. This allows the model perform better on back-and-fourth conversational tasks. DialoGPT comes in three sizes: small, medium, large. Sample text from the large model suggests that it would have performed much better, but we were only able to train the small model due to Colab memory limits. The small DialoGPT model is not very robust and often repeats itself.

Our approach to hyperparameter selection was guided by HuggingFace's default hyperparameters, the GPT2 training guide mentioned previously[3], past experience, and performance on the validation set. First, we knew that the Adam optimizer performed well on GPT2-like models because of our experiences with past projects. Next, we tried various hyperparameter selection techniques like grid search over learning rate values and epsilon levels. i.e. For each [epsilon, learning rate] pair, we initialized a separate model then trained and evaluated it over many epochs of our training and validation sets. We then took the best set of hyperparameters from this search and experimented with the number of epochs. In the end, we were unable to find better hyperparameters than many that we started with, so we decided to shift our attention to other facets of ours model architecture (e.g. experimenting with different architectures, preprocessing methods, etc.).

### 6.1 Core model

The first architecture we consider was a language model with a basic causal mask, as shown in Figure 1b. As mentioned above, the model was pre-

[1] https://github.com/smeyerhot/CS685

[2] http://reyfarhan.com/posts/easy-gpt2-finetuning-huggingface

[3] http://reyfarhan.com/posts/easy-gpt2-finetuning-huggingface

trained on DialoGPT and was fine-tuned on our preprocessed dataset.

The second model we consider is a prefix language model, shown in Figure 1c. The only difference between the prefix language model and the regular language model is in the attention mask. For the prefix language model, the tokens of the prefix can attend to all other tokens in the prefix, and the rest of the tokens use a causal mask. The regular language model uses only a causal mask. Since the prefix language model is not implemented in HuggingFace, we made a fork of HuggingFace/transformers and formed the causal prefix masks ourselves.[4]

Finally, we tried making an encoder-decoder model, shown in Figure 1a, using HuggingFace's EncoderDecoderModel. We used code from our other models and looked at the usage examples from HuggingFace.[5] The encoder-decoder model was much larger than the decoder only models and was therefore difficult to load into Colab's memory using the same pretrained DialoGPT model. If we used a batch size of one, we were able to load it into memory most of the time, but sufficiently tuning the hyperparameters proved difficult for this reason. We were able to get the encoder-decoder model to produce either text that made sense but was not related to COVID-19 or text that was nonsensical but related to COVID-19. Overall, the encoder-decoder model performed much worse than either the language model or prefix language model in preliminary experiments, so we decided to leave it out of the final analysis and use the two better models.

### 6.2 Similarity Model

As an extension of the core models' training approach described in Section 6.1, we also built a similarity model that retrieves the most semantically similar question-answer pair from the training set and prepends the pair to the input of the core models, as shown in Figure 2.

We extracted the embeddings of the patient-doctor conversations in the training set from the second-to-last hidden layer of HuggingFace's BioBERT.[6] In this step, we averaged the BioBERT's second-to-last hidden layer of each token to produce a single vector of length 768. Thus, we were able to compute and extract the final sentence embedding of each $p_i$ and $d_i$ from our preprocessed data.

After creating a training split with 75% of the entire dataset, we retrieved the most similar pair of $p_i$ and $d_i$ to each input question from the training dataset. For each $p_i$ in the training dataset, we performed semantic similarity search to retrieve the most similar answer $d_{sim}$ and the corresponding question $p_{sim}$ in the training dataset. To do so, we used the BioBERT sentence embeddings of all $p_i$ and $d_i$ in the training dataset. The similarity search algorithm used cosine similarity:

$$\text{Cosine similarity (A,B)} = \frac{A \cdot B}{||A|| \times ||B||},$$

where A and B are real-valued vectors and the value ranges from -1 (perfectly dissimilar) and 1 (perfectly similar). To calculate the cosine similarity, we used FAISS (Johnson et al., 2017) library developed by Facebook AI Research Group that provides tools for efficient similarity search with the support of GPU implementation. Thus, we conducted FAISS similarity search using the extracted BioBERT embeddings of each answer $d$ in the training dataset and the embeddings of the current $p_i$. To implement the FAISS search, we looked at an example on GitHub that also used FAISS cosine similarity function for semantic search. [7].

Once we obtained the most similar question-answer pair $(p_{sim}, d_{sim})$ in the training dataset, we concatenated the retrieved question-answer pair to the original input conversation. For example, a conversation with just one patient question $p_0$ would look as follows:

$$p_{sim} d_{sim} p_0,$$

where $p_{sim}$, $d_{sim}$ and $p_0$ have already <patient> and <doctor> in themselves.

Finally, we provide the newly formatted input of the original question prepended with the most similar question-answer pair to fine-tune DialoGPT and generate the next predicted answer. Note that for validating and testing the trained model, we used the original format of inputs from the core models without retrieving the most similar pair, as our objective of the similarity model is to observe whether providing more context to DialoGPT improves the quality of generated answers.

We applied the strategy of similarity model on both a language model with a basic causal mask

[4] https://github.com/kteavery/transformers

[5] https://huggingface.co/patrickvonplaten/bert2gpt2-cnn_dailymail-fp16

[6] https://huggingface.co/dmis-lab/biobert-base-cased-v1.1

[7] https://github.com/re-search/DocProduct/blob/master/notebooks/FaissEvalTopKColab.ipynb

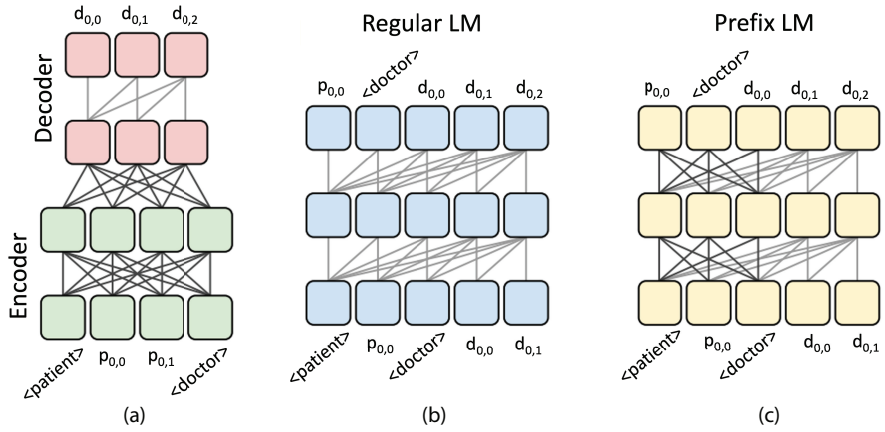


Figure 1: The core architectures considered in the paper. This figure is a modified version of Figure 4 in Raffel et al. (2019). In this example, $p_0$ is a patient question and $d_0$ is the doctor's response that we are trying to predict. The second subscript is the index of the token. (a) an encoder-decoder model, (b) a language model with a basic causal mask, (c) a prefix language model.

and a prefix language model. Overall, the similarity model with a causal mask performed better than the similarity model with a causal prefix mask. The latter generated text with more repetitions and was more nonsensical in preliminary experiments. Therefore, we decided to choose the similarity model with a basic causal mask for our final analysis.

On Github,[8] the regular language model is implemented in decoder_only.ipynb; the prefix language model is in prefix.ipynb; and the similarity model is in biobert-decoder.ipynb. The encoder-decoder model is in encoder_decoder.ipynb, but this model was preliminary and not used in the final analysis.

## 7 Results and Error Analysis

We ran the regular language model, prefix language model, similarity model, and baseline five times each in order to account for variation. The regular and prefix languages models are the same, except the first uses a causal mask and the second uses a causal prefix mask. The regular language model performed better, so the similarity model also uses a regular causal mask.

Figure 3 shows the loss curves for the regular language model, prefix language model, and similarity model. The saved models used for evaluation used whatever epoch had the lowest validation loss. Oddly, the first and sometimes second epoch of the validation loss often seemed lower than the training loss. Upon further inspection, this was because the validation loss was measured after the training loss was completed. For example, the training loss at epoch 1 may start high at 5.0 and drop to 0.8, and the validation loss measured after epoch 1 may be 0.9. However, the average training loss for epoch 1 might be 1.4 because the loss without any training was so high.

### 7.1 Automatic Evaluation

The best way to analyze the model output is by hand, but we have also included some automatic evaluation measures, including the f-score, precision, and recall for ROUGE-1, ROUGE-2, and ROUGE-L (Lin, 2004). ROUGE-1 and ROUGE-2 look at the the overlap in unigrams and bigrams, and ROUGE-L looks at the longest common subsequence (LCS) in the generated text and the reference. The f-score is calculated as follows:

$$F = 2 \times \frac{precision \times recall}{precision + recall}.$$

The recall measures the unigrams/bigrams/LCS in the reference text that appear in the generated text, and the precision measures the unigrams/bigrams/LCS in the generated text that appear in the reference text.

Figure 4 plots the average precision, recall, and f-score with 95% confidence intervals. Some error bars are difficult to see if the bounds are tight. Overall, the similarity model had high precision and low to average recall compared to other models. The similarity model's output was very lengthy compared to the reference text and other models' generated text. Many words in the gen-

[8] https://github.com/smeyerhot/CS685

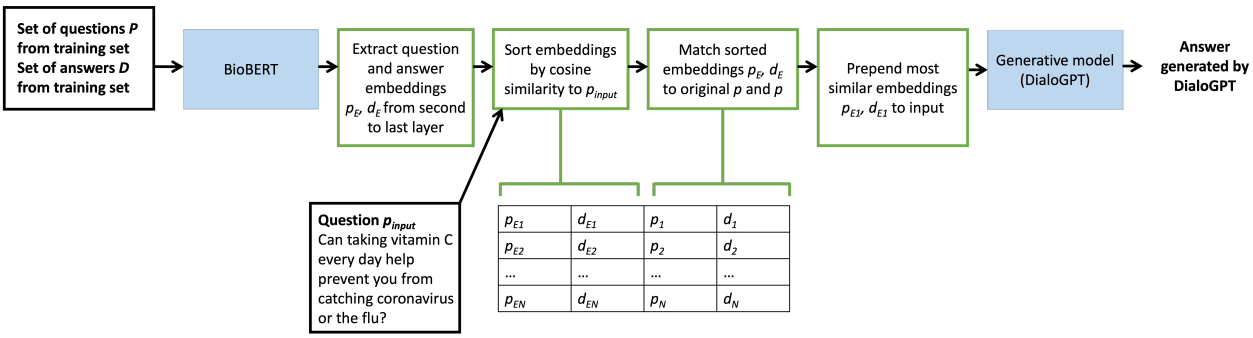


Figure 2: The pipeline for the similarity model.

erated text of the similarity model appeared in the reference. The regular language model performed better than the prefix and baseline models, which performed the worst. These results will be contextualized in the following section.

### 7.2 Hand Evaluation

We used two rating scales for our hand evaluations. The first is shown in Table 2 and measures the fluency of the answer text (Iyyer, 2021). The second scale, Table 3 measures how well the generated text answers the posed question. The scale is based on the rating scale in Oniani and Wang (2020), though we made slight modification. If a generated answer makes logical sense and is related to the question and COVID-19, *but* it gives bad medical advice, then it gets downgraded from a four to at most a score of three. For example, the sentence, "take antibiotics for COVID" makes logical sense (i.e., it is possible to do), but it is terrible advice. An answer with a rating of four may not be perfect, but it requires accuracy.

For the hand evaluation, our group discussed the criteria and provided example answers for each score. Despite this, there was definitely disagreement among annotators. The questions, ground truth answers, and generated answers were typically the length of a paragraph, and the generated answers would often contain some high-quality content and some low-quality content. Further, the user-generated questions were not always clear, so annotators would sometimes interpret the questions differently. Finally, we are not medical experts, so evaluating the quality of a plausible-seeming answer occasionally proved challenging. To resolve this issue, we would see how well the generated answer matched the recommendation in the ground truth, and if necessary, we would look up relevant medical information to see if a recommendation made sense. Overall, we erred on the side of caution and only rated complete and sound answers with a score of five.

Each example was rated by two annotators. After discussion, we decided that averaging the two annotator scores best accounted for disagreements. Typically, an argument could be made for each rating, so we wanted to take both scores into account in the final score. We also calculated the average Cohen's kappa with linear weights, which accounts for the amount of difference in two scores. e.g., Scores of 3 and 5 are more similar than scores of 1 and 5.

All four models were evaluated on the same test set of 144 examples. To account for variability, the first 28 examples were drawn from the first run of each model, and the second 28 examples were drawn from the second run of each model, etc. The average final fluency and question scores can be found in Table 4, along with the average Cohen's kappa for both the fluency and question scores.

Unsurprisingly, the baseline was the most fluent model because it pulled answers directly from the training set, so the answers were all written by humans. However, they rarely answered the question and thus had an average question score of 3.01. The regular language model seemed to answer the questions slightly better, but it was less fluent. The prefix model performed worse on both counts. Finally, the similarity model got average scores between the regular and prefix language models.

Because the baseline lifted answers from the training set, they were generally factually correct and related to COVID-19 or respiratory diseases. However, they rarely answered the question being asked. On the other hand, the models used for our approach were more likely to give an answer related to the question, though they were less likely to give factually correct responses. (Factually in-

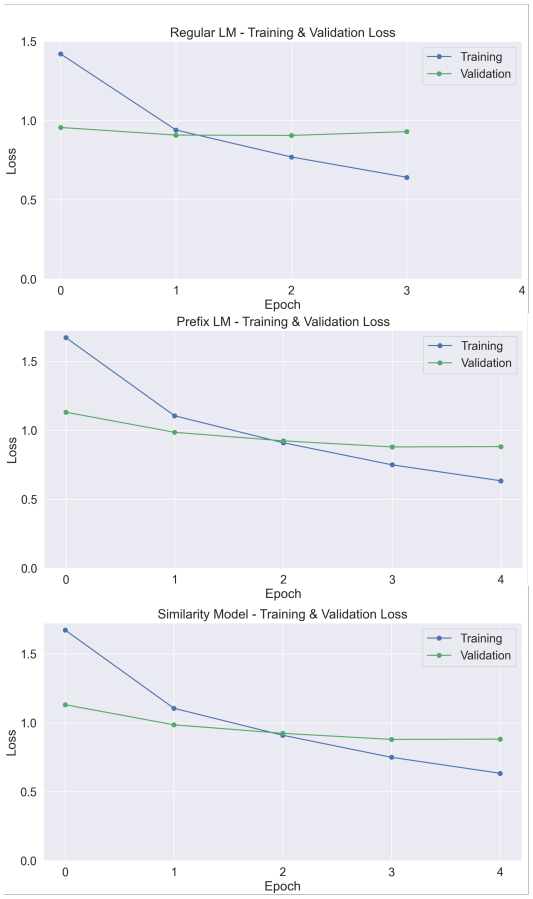


Figure 3: The training and validation loss curves for the regular language model (top), prefix language model (middle), and similarity model (bottom). The training loss is shown in blue and validation loss is shown in green. The regular language model never benefited from having five epochs, which is why it is graphed with four epochs.

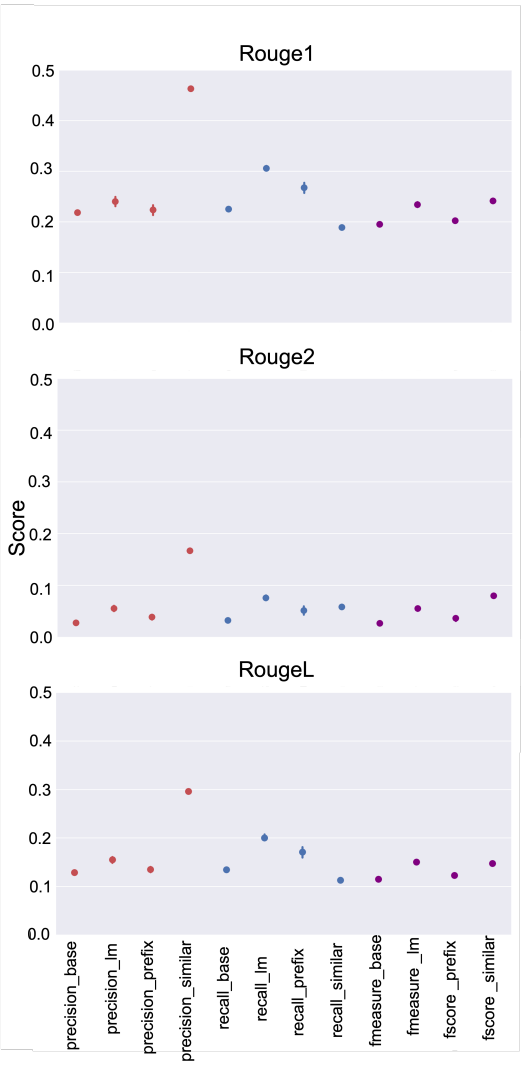


Figure 4: ROUGE score results. The average precisions (red), recalls (blue), f-scores (purple) were plotted with 95 percent confidence intervals for all four models.

correct responses got a score of 3.) Occasionally, our models would give patient-like output rather than doctor-like output. For example, the model might ask more questions rather than answer the input question. Patient-like responses could get a score between 1 and 3, depending on how much logical sense they made.

The regular language model was generally less fluent than the baseline, and it often tried to answer the question but gave bad medical advice. Sometimes, it would partially answer the question. There was some amount of disagreement among annotators because there were many borderline cases. For example, if a model gives some good advice and some unrelated advice, one annotator might rank the generated text as a 4, while another might rank it as a 3. Further, the regular language model would sometimes generate perfect English and non-native or disfluent English in the same response. Annotators would sometimes rank these differently. We decided that an average of the two scores account for both arguments.

The prefix language model tended to give lower-quality output compared to the regular language model. We suspect that forcing the model to predict prefix tokens was useful during training, so the prefix causal mask actually hurt performance. Our fine-tuning dataset is very different from the data that DialoGPT was trained on, so predicting the prefix may have been helpful. Annotators substantially agreed on the quality of the prefix model's output because it consistently rarely answered the question but was generally on-topic. The fluency was consistently disfluent or

| | Fluency Score |
|---|---|
| 5 | flawless English |
| 4 | good English |
| 3 | non-native English |
| 2 | disfluent English |
| 1 | incomprehensible |

Table 2: A five-point rating scale that measures the fluency of the model's answer (Iyyer, 2021).

non-native English.

The results of the similarity model were interesting. Often, the model would give an adequate or even good answer in the first half of the response, but the latter half would contain unrelated patient-like text and sometimes nonsensical sentences. In the latter half, the model would sometimes ask and answer a completely different question. The fluency also degraded between the first and second half. This caused quite a bit of disagreement among annotators, as some annotators gave the model credit for giving the right answer, and others would harshly penalize the model for also generating patient-like and unrelated text. We decided that an average of these two responses was most appropriate.

We believe that because the similarity model had another patient-doctor conversation prepended, it would generate another conversation after it responded to the input question. However, the similarity model definitely improved the quality of the generated answer before it started generating unrelated text.

The models tended to perform better on COVID-19-related questions, which were about half the dataset. The other half of the dataset was about respiratory diseases from the MedDialog dataset. The MedDialog examples generally required the model to have more medical knowledge, while the COVID examples only required the knowledge of one disease.

## 8 Contributions of group members

List what each member of the group contributed to this project here. For example:

- Teodoro: Worked on preprocessing, model building, and augmentation. Wrote baseline model. Helped Minhwa troubleshoot BioBERT embedding issues (like exploring how to merge two vocabularies). Worked on various utility functions.

- Kate: Worked on preprocessing, model building, and augmentation. Worked on ways to make the model better (like adding special doctor/patient tokens and experimenting with different model types). Worked on code for evaluation.

- Minhwa : Worked on extracting embeddings of question-answer (QA) pairs from BioBERT. Implemented cosine-similarity function by using FAISS library. Structured and implemented a framework of similarity model.

- Juhyeon: Reran the prefix language model and similarity model for the final analysis. Helped Minhwa for the implementation and hyperparameter tuning of similarity model.

Everyone participated in writing the final report, and everyone worked on hand evaluations and examined the models' output for the error analysis.

| | Question Score |
|---|---|
| 5 | The question is generally answered. |
| 4 | The answer is somewhat related to the question and COVID-19 (or respiratory disease), and it makes logical sense. |
| 3 | The answer is not related to the question. The answer is about COVID-19 (or respiratory disease) and makes logical sense, OR it answers the question and makes sense but gives bad advice. |
| 2 | The answer is somewhat related to the question or COVID-19 (or respiratory disease). It does not make logical sense. |
| 1 | The answer is unrelated to the question and does not make sense. |

Table 3: A five-point rating scale that measures the adequacy of the model's answer (Oniani and Wang, 2020).

| Model | Question Score | Fluency Score | Question Cohen's | Fluency Cohen's |
|---|---|---|---|---|
| baseline | 3.01 | 4.44 | 0.409 | 0.330 |
| regular language model | 3.18 | 3.60 | 0.227 | 0.222 |
| prefix language model | 2.70 | 2.90 | 0.771 | 0.672 |
| similarity model | 2.92 | 3.13 | 0.062 | 0.142 |

Table 4: The results of the hand evaluation. The left side of the table shows the average scores for each model, and the right side shows how much the evaluators agreed.

## 9 Conclusion

In conclusion, our models tend to COVID-related answers to patient questions, but it does not give the sound medical advice that would be given by a real doctor of medicine. Before beginning the project, we realized that open-domain medical question-answering on real-world conversational data was a difficult task, so we were not surprised by our results. As mentioned in the proposal, we were only implementing a few components of a medical question-answering chatbot in our prototype, which would normally involve a very sophisticated information retrieval system. Given the difficulty of the task, our computational resources, and the limitations of our methods, we were overall satisfied with our results.

We were surprised by how difficult it was to teach the models their conversational role. In preliminary experiments, the models almost always produced patient-like output, rather than doctor-like output. Adding special tokens to separate the patients' words from the doctors' words was important to teaching the model its conversational role.

### 9.1 Potential Improvements

If we could continue working on the project in the future, we would be more discriminating about the examples from the MedDialog dataset that we use in our final dataset. We pulled examples related to respiratory disease but did not manually read through every example. If we had more time, we would carefully choose each example to ensure it was as related to respiratory viral diseases as possible.

If we had more time and computational resources, we would also train on a larger pretrained model and explore more sophisticated information retrieval techniques. For example, due to time limits on Colab's GPU memory access, we could not retrieve only one question-answer pair that has the most closest distance with the BioBERT embedding of the given question. We expect that retrieving more QA pairs as a context to the similarity model would improve the quality of generated answers from the fine-tuned DialoGPT.

Finally, we would split the "question score" rating scale into 1) a scale that measures the accuracy of the information and 2) a scale that measures how on-topic the response is. The question score scale from Oniani and Wang (2020) seemed as though it was trying to measure several aspects of the answer on one scale.

### 9.2 Ethical Implications

The models outlined in this project should not be used to give actual medical advice. Generative text can give highly unreliable output, making it unsuitable for real-world deployment. The purpose of this project was to explore open-domain question answering on real conversational data, not to give real medical advice.

## References

Alsentzer, E., Murphy, J., Boag, W., Weng, W.-H., Jindi, D., Naumann, T., and McDermott, M. (2019). Publicly available clinical BERT embeddings. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 72–78, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Beltagy, I., Lo, K., and Cohan, A. (2019). SciBERT: A pretrained language model for scientific text. In *Proceedings*

*of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Cer, D., Yang, Y., Kong, S.-y., Hua, N., Limtiaco, N., John, R. S., Constant, N., Guajardo-Céspedes, M., Yuan, S., Tar, C., et al. (2018). Universal sentence encoder. *arXiv preprint arXiv:1803.11175*.

Chen, S., Ju, Z., Dong, X., Fang, H., Wang, S., Yang, Y., Zeng, J., Zhang, R., Zhang, R., Zhou, M., Zhu, P., and Xie, P. (2020). Meddialog: a large-scale medical dialogue dataset. *arXiv preprint arXiv:2004.03329*.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Iyyer, M. (2021). Lecture notes from advanced natural language processing.

Johnson, J., Douze, M., and Jégou, H. (2017). Billion-scale similarity search with gpus. *arXiv preprint arXiv:1702.08734*.

Ju, Z., Chakravorty, S., He, X., Chen, S., Yang, X., and Xie, P. (2020). Coviddialog: Medical dialogue datasets about covid-19. *https://github.com/UCSD-AI4H/COVID-Dialogue*.

Lee, J., Yoon, W., Kim, S., Kim, D., Kim, S., So, C. H., and Kang, J. (2019). BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Lin, C.-Y. (2004). ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Oniani, D. and Wang, Y. (2020). A qualitative evaluation of language models on automatic question-answering for covid-19. In *Proceedings of the 11th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics*, BCB '20, New York, NY, USA. Association for Computing Machinery.

Peng, Y., Yan, S., and Lu, Z. (2019). Transfer learning in biomedical natural language processing: An evaluation of BERT and ELMo on ten benchmarking datasets. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 58–65, Florence, Italy. Association for Computational Linguistics.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2019). Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. pages 1–53.

Safi, Z., Abd-Alrazaq, A., Khalifa, M., and Househ, M. (2020). Technical aspects of developing chatbots for medical applications: Scoping review. *J Med Internet Res*, 22(12):e19127.

Salton, G., Wong, A., and Yang, C. S. (1975). A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620.

Zeng, G., Yang, W., Ju, Z., Yang, Y., Wang, S., Zhang, R., Zhou, M., Zeng, J., Dong, X., Zhang, R., Fang, H., Zhu, P., Chen, S., and Xie, P. (2020). MedDialog: Large-scale medical dialogue datasets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9241–9250, Online. Association for Computational Linguistics.

Zhang, Y., Sun, S., Galley, M., Chen, Y.-C., Brockett, C., Gao, X., Gao, J., Liu, J., and Dolan, B. (2020). Dialogpt: Large-scale generative pre-training for conversational response generation. In *ACL, system demonstration*.