GNG 5125 Data Science Applications
Final Project

Medical recommendations powered by scispaCy based on Medical Question-Answer Datasets

Thomas Courtney (8335223)
Justin Dalrymple(300100350)
Haowei He (300158003)

July 2022

Contents

1 Introduction **5**

2 Methodology **6**

3 Data Description & Datasources **7**
3.1 Overview 7
3.2 Pre-Processing 8
3.2.1 MedInfo 8
3.2.2 MedQuAD 9
3.2.3 LiveMed 11
3.3 Data Trends and Anomalies 11

4 System Design **13**
4.1 Model Pipeline 14
4.2 Answer Pipeline 15

5 Evaluation **16**

6 Discussion **16**

7 Conclusion **16**

List of Figures

1 Traditional vs Proposed medical information pipeline for low risk questions. Thicker arrows indicate higher traffic 5
2 Type distribution within final dataset 12
3 Focus distribution within final dataset limited to those with greater than 10 occurances 12
4 Overview of recommendation system 14
5 Alternative clustering pipelines 15

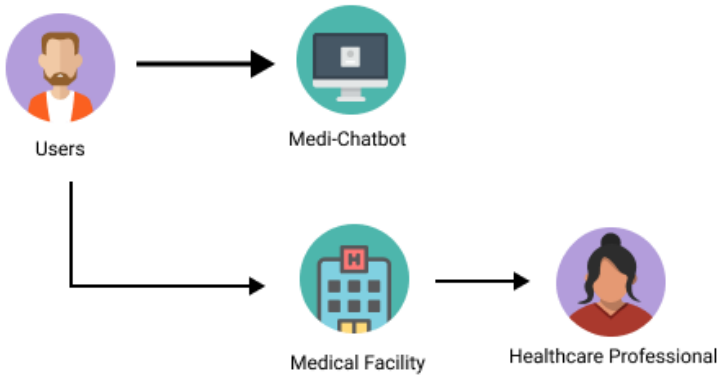
List of Tables

1 Selected Q/A Medical Datasets 7
2 Desired Q/A structure 8
3 Available Question Types 9
4 MedInfo Question Type Mapping 10
5 MedQuAD Question Type Mapping 10
6 LiveMed Question Type Mapping 11

1 Introduction

Medical Recommendation Systems offer a unique way to relieve stress on the already stretched-thin medical infrastructure we have today. Coming off the tail end of the COVID-19 pandemic, many medical providers have reported a lack of staff, and lack of resources [4,5] drastically reducing their effectiveness in offering care to the various persons in need. One way to redirect some of the pressure on these systems is to effectively offer a triage system that could offer persons basic information to their various medical questions without needing to go directly to a provider.

(a) Traditional Medical Information Pipeline



(b) Proposed Medical Information Pipeline

Figure 1: Traditional vs Proposed medical information pipeline for low risk questions. Thicker arrows indicate higher traffic

This triaging system would operate like a chatbot, presenting the user with a simplistic user interface to ask their medical-related questions. In the report below, we describe and evaluate a system that accomplishes just this by leveraging a name entity recognition library developed explicitly for analyzing biomedical terminology known as scispacy [6] and question clustering to determine the closest answer or answers to a users question.

For completeness, we also outline this application's limitations and future improvements that could improve its scope and support further.

2 Methodology

(This section is adapted from our proposal as it remains relevant)

There are four components required for a successful recommendation engine:

1. Adequate understanding of the question - What are the key points asked
2. Sufficiently clustered questions - What are the related questions
3. Sufficient data with proper labelling
4. Simplified interaction between the questioner and the engine

Undertaking the first component will be through leveraging scispaCy [6]. scispaCy builds upon the success of spacy, a Natural Language Processing library that excels at information extraction of text but has its focus on the scientific-based vernacular. This will be fundamental for creating a semantical understanding of medical terminologies asked by the user.

The second component builds off the first but for the determining overall similarities in questions. Since each question could potentially match a variety of answers, especially for questions not seen before, we will cluster similar questions together. This will allow us to quickly return related answers to questions regardless of exact matches.

The third component will be fulfilled by utilizing a combination of medical QA datasets (outlined in the following section) which contains many medical questions and answers similar to the questions most likely to be asked by a user.

The fourth and final component would be accomplished by developing a simple user interface that allows users to enter their questions and see the recommended response. We will investigate various interfaces for this application to present an intuitive interaction for the user. This could be a "google search" or a chatbot implementation.

In terms of workflow, here is how our system will work under the hood:

1. Receive input via the user

2. Clean user input
3. Tokenize cleaned input
4. Use a comparison method to find the nearest answer cluster
5. Return ordered answers based on closest to furthest in matching similarity.

3 Data Description & Datasources

3.1 Overview

A sufficient dataset of common questions is paramount to respond to a variety of questions adequately. Initially, we hoped to utilize a comprehensive EMR dataset that provided substantial medical information pertaining to medications and various ailments. However, the dataset in question, IBM EMRQA [7], only contained rough information about a patient's medical condition and colloquially abbreviated information which did not lend itself to a question-answer framework. As such, we selected a collection of Medical Q/A datasets outlined in table 1

Selected Datasets			
Name	Number of Q/A Pairs	Publish Year	Ref
MedInfo	689	2019	[3]
MedQuAD	2479	2019	[2]
LiveQA MedicalTask Q/A	550	2017	[1]

Table 1: Selected Q/A Medical Datasets

Together, this provided us with 3718 raw question-answer pairs. While this is relatively small, there is a potential to increase this overall dataset with additional parsing of the MedQuAD source. In their publication, they report a total dataset of 47,457 QA pairs, however, many were removed due to copyright conflicts. The answers are still available but would require intensive scrapping, which was out of scope for this project.

3.2 Pre-Processing

The above datasets were first standardized to ensure our processing steps could effectively and consistently parse the incoming data. The desired structure was in the form outlined in Table 2, where the list of types is outlined in Table 3

Q/A Datastucture		
Name	Datatype	Description
question_raw	String	The raw question as presented by the dataset
question_proc	String	A processed question obtained by normalizing the data via spacy
question_words	[String]	The processed question in word segments
types	[Type]	A list of applicable types parsed from the raw question text
focus	String	An applicable focus entity parsed from the raw question text
answer	String	The raw answer as presented by the dataset

Table 2: Desired Q/A structure

The type and focus information were parsed differently depending on the dataset and will be described in greater detail in the following sections.

Regardless of the dataset, the parsed focus data was cleaned by lowercasing the text and removing non-alpha text, multiple white spaces and new line delimiters. Similarly, the answer texts were also cleaned lightly by removing multiple white spaces and new line delimiters.

Additionally, for each pair, we excluded those whose types we could not parse and who did not have an answer connected to the question. This resulted in an overall reduction of our dataset to 2323 pairs.

3.2.1 MedInfo

To determine the focus for the MedInfo Q/A dataset, we selected each pair's Focus(Drug) column. We then normalized it using spacy (lemmatization, removal of stopwords and punctuation) and passed it to our additional cleaning function for consistency.

The types were determined using the 'Question Type' property from the pair and mapping it to the corresponding label for our desired structure.

Q/A Types	
Name	Description
Usage	How some medical device, medication, etc should be used
Composition	What a medical-related substance is made of
Alternative	Any alternatives to the queried medical substance
Appearance	Descriptive appearance of the queried medical substance
Dosage	Dose-related information for the queried medical substance
Interaction	Interactions the queried medical substance could have with other substances
Side Effect	Negative interactions the queried medical substance could have with the user
General	All informational data that does not fall within the previous categories

Table 3: Available Question Types

For the mapping table see Table 4. In this dataset, only one type would be parsed, which was assigned to the primary_type property of our Q/A pair data structure.

After processing, our MedInfo dataset went from 689 to 664 pairs

3.2.2 MedQuAD

To determine the focus for the MedQuAD Q/A dataset, we parsed the raw question text using the spacy NLP function. After which, we selected the first entity from the list. We then passed it to our additional cleaning function for consistency.

The types were determined using mapping in Table 5 where the presence of any of the key terms outlined in the key terms column inside the raw question text resulted in the connected type label. Though multiple types could be parsed in this dataset, we selected the first one as the primary_type property of our Q/A pair data structure.

After processing, our MedQuAD dataset went from 2479 to 1555 pairs

Q/A Types	
Label	Keyterms (lowercased)
Usage	'usage', 'usage/time', 'stopping/tapering'
Composition	'ingredient'
Alternative	'alternatives', 'brand names', 'comparison'
Appearance	'appearance'
Dosage	'dose', 'dose/potency'
Interaction	'interaction', 'contraindication'
Side Effect	'side effects', 'overdose', 'forget a dose', 'stopping/side effects'
General	'information', 'indication', 'action', 'action/time','pronounce name', 'availability', 'time/duration', 'action/effectiveness', 'storage and disposal', 'manufacturer'

Table 4: MedInfo Question Type Mapping

Q/A Types	
Label	Keyterms (lowercased)
Usage	'used', 'use'
Composition	'ingredients', 'made of'
Alternative	'alternatives', 'substitute'
Appearance	'smells like', 'looks like', 'taste like'
Dosage	'dose', 'dosage'
Interaction	'interact', 'used with'
Side Effect	'side effects', 'adverse effects', 'reactions', 'cause'
General	'what is', 'should i know', 'what are'

Table 5: MedQuAD Question Type Mapping

3.2.3 LiveMed

We parsed the 'FOCUS' annotation property from the QA pair to determine the focus for the LiveMed Q/A dataset and selected the first item. We then passed it to our additional cleaning function for consistency.

The types were determined using the 'TYPE' annotation property from the pair and mapping it to the corresponding label for our desired structure. For the mapping table see Table 6. In this dataset, only one type would be parsed, which was assigned to the primary_type property of our Q/A pair data structure.

Q/A Types	
Label	Keyterms (lowercased)
Usage	'usage', 'tapering'
Composition	'ingredient'
Alternative	'alternative', 'comparison'
Appearance	None
Dosage	'dosage'
Interaction	'effect', 'interaction', 'contraindication'
Side Effect	'side_effect', 'complication'
General	'treatment', 'prevention', 'diagnosis', 'cause', 'information', 'symptom', 'storage_disposal', 'action', 'susceptibility', 'indication', 'lifestyle_diet', 'prognosis', 'person_organization'

Table 6: LiveMed Question Type Mapping

After processing, our LiveMed dataset went from 550 to 104 pairs

3.3 Data Trends and Anomalies

Out of all the pairs, there was a noticeable difference in the types of questions asked, as seen in Fig 2, particularly skewed to 'General' questions. This could be explained by improper labelling, but further analysis would be required to verify this source of error.

Concerning the focuses parsed from the pairs, there were 936 unique values. This makes sense as each focus is the primary entity of the question. This also shows that almost half of the pairs were related in some fashion (2000 pairs, only 936 focuses). The most common focuses were symptoms and treatments, accounting for roughly 1/3 of the entire dataset.

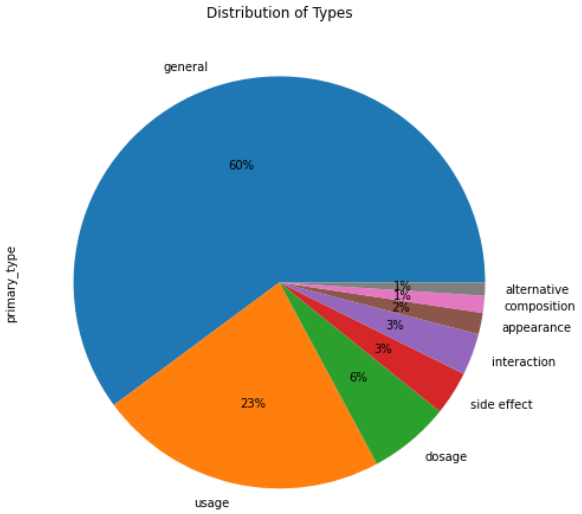


Figure 2: Type distribution within final dataset

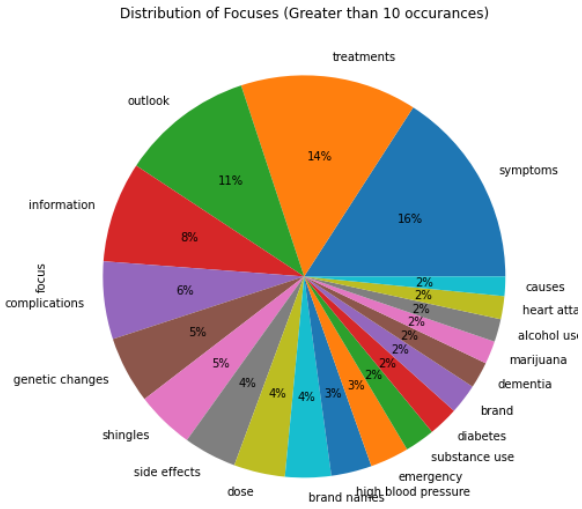


Figure 3: Focus distribution within final dataset limited to those with greater than 10 occurances

When looking at the focus parsing, there were some apparent errors in what was selected. Of these errors, they could be distilled to two categories:

1. Ambiguous single/double characters
2. Ambiguous terms
3. Type duplication

Ambiguous single/double characters referred to instances where the plural indicator was separated from the word, such as 'swimmer s ear,' 'schmorl s nodes' and 'traveler s', occurrences of single characters such as 'b' or 'u' or the single characters themselves are ambiguous within the context of the selection such as in 'n a exploration'. Some of these could be filtered out, for example, by excluding single character focus values. However, others are more difficult because they provide context, such as in 'vitamin a' or 'hepatitis c'. These single-letter occurrences could also be an indication of poor processing during the cleaning methods, though further investigation would be needed to verify this claim.

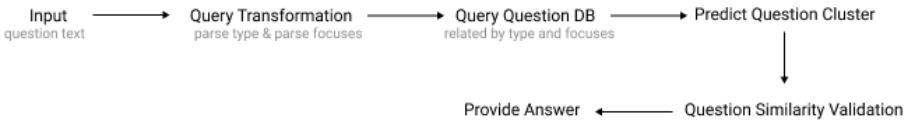
Ambiguous terms include words that were longer than one character but too ambiguous to provide context. For example 'odd', 'group', 'questions' and 'near'. Similar to the single character occurrences, these words most likely exclude other words providing context and would have to be analyzed further to determine why the full selection was not included.

Finally, type keyword duplication was also discovered in the focus data, commonly with terms like 'information,' 'side effects,' 'symptoms,' and 'brand names. Each of these is used to determine the higher-order type of the question and should not also be the focus of the question itself. Skipping such occurrences when looking for the focus of the question could be achieved by excluding these words when parsing.

4 System Design

Our medical recommendation system was designed to satisfy the primary tasks outlined in the methodology, providing the user with an adequate answer to their medically related questions. We wanted to support the most extensive breadth of medical questions but also focus on more common ones. To this end, our data selection successfully contained a greater selection of general questions. However, the process of returning the correct answer to these questions depended on how we designed our data pipeline.

(a) Model Pipeline



(b) Answering Pipeline

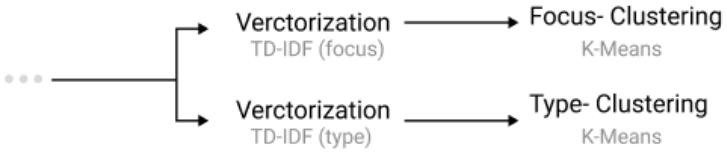
Figure 4: Overview of recommendation system

4.1 Model Pipeline

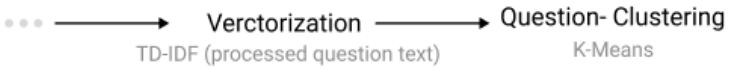
After the initial preprocessing of the data, as outlined in the Data Description & Datasources section, we selected two fields to act as the key identifier for the eventual clustering. We decided on these two categories instead of looking at the question alone due to the duplication of terms in the question and a vastly greater number of computationally intensive clusters. Since we already categorized the questions into their type and general focus, the usefulness of the whole question was relegated to the last step for similarity discussed in the Answer Pipeline section.

With the identifier selected, we vectorized this column and performed a K-means clustering on the resultant series. The number of clusters was capped at 1000. Since there are 900 focuses and eight types, ideally, there should be a cluster for each combination, however such a design, when tested, was computationally taxing. Thus we limited our clusters to 1000, with an eventual plan to redesign how we cluster said categories. We also investigated the elbow method to determine the optimal amount of clusters, but this too never reached a global minimum before 1000. Based on our general understanding that such a minimum would exist around 1800, this too made sense.

We also investigated an alternative clustering pipeline that separated clustering into two steps; once for the type and once for the focus. This would reduce the number of items in the second clustering and should reduce computation time. Unfortunately, we left this out of the final code due to time constraints. We want to revisit this in the future to compare the performance.



(a) Two Leveled Clustering with Seperated Focus and Type



(b) Processed Question Text Clustering

Figure 5: Alternative clustering pipelines

4.2 Answer Pipeline

Following a similar process, the Answer pipeline aims to determine the correct focuses and type from the question text. After which, it gathers a list of potential questions matching the various combinations of focus and type. In this step, we select all questions from our questions database that have the same type and any of the focuses found in the focus parsing. This allows us to gather a broader range of topics for the initial cluster prediction.

This subset of questions is used to generate the cluster key (focus+type) fed to our model to predict the correct cluster. After the cluster is predicted, we perform a similarity check between all the samples within this cluster and the original question text query given by the user. We select the question pair with the highest similarity to obtain the answer.

While this is generally sufficient, we also wanted to account for instances where our system made the wrong prediction. As such, we limit the similarity value to those greater than 0.8. If the max similarity is less than this value, we return a message "Sorry, we couldn't find any information to match your question," which is preferred over wildly incorrect information being provided to the user.

5 Evaluation

6 Discussion

Taking the preprocessed text data, the feature engineering model TF-IDF was chosen, quantifying the importance of string representations in a document amongst a collection of documents.

We then took the normalized, preprocessed data, which then would be utilized for a clustering algorithm. We chose the clustering algorithm K-Means, as the goal of such an algorithm is to check for similar data and cluster them together while trying to separate each cluster as far as possible. In relation to our recommender system, K-Means is the optimal algorithm since when a medical question is asked, our system returns the most relevant answer in theory.

Finally, from our provided code, using the K-Means algorithm, a function was created which returns the most relevant answer to the question asked. Using similarity matching, if the said answer had a matching value of 0.8 or greater, this states the answer is related relative to the question, and the answer would be returned. However, if the matching value has a value below 0.8, no answer would be returned as there was insufficient information to match the question.

7 Conclusion

After many iterations and tests with our model, using medical data from MedInfo, MedQuAD, and LiveMed a dataset was derived in hopes of providing users with accurate answers to their medical questions. The data from these three sources were then preprocessed in a variety of ways such as lemmatization along with the removal of stop words and punctuation using the spaCy library, also the dataset was passed through a custom-built cleaning function which was tailored to our dataset in efforts for the most optimal dataset for our recommender system. Additionally using feature engineering, the TF-IDF model was chosen for our system since this model quantifies importance based on string representations. Finally, we can see that our recommender system was successfully built, it allows users to enter their medical questions and see the recommended response by our system. This system works by using the three data sources previously mentioned. As we already know the medical industry has been stretched thin, due to the recent pandemic along with an aging population. By building this rec-

ommender system we hope to provide users access to medical information in a quick and stress-free manner, but at the same time allow for some relief for the medical staff who are already overburdened in their current state.

References

[1] Asma Ben Abacha, Eugene Agichtein, Yuval Pinter, and Dina Demner-Fushman. Overview of the medical question answering task at trec 2017 liveqa. In *TREC 2017*, 2017.

[2] Asma Ben Abacha and Dina Demner-Fushman. A question-entailment approach to question answering. *BMC Bioinform.*, 20(1):511:1–511:23, 2019.

[3] Asma Ben Abacha, Yassine Mrabet, Mark Sharp, Travis Goodwin, Sonya E. Shooshan, and Dina Demner-Fushman. Bridging the gap between consumers' medication questions and trusted answers. In *MEDINFO 2019*, 2019.

[4] Yasmine Hassan. Health-care workers call for government help as burnout worsens and staff shortages increase | cbc news, Jun 2022.

[5] Akshay Kulkarni. We looked at data on temporary closures, reduced services in b.c. hospitals this year. here's what we found | cbc news, Jul 2022.

[6] Mark Neumann, Daniel King, Iz Beltagy, and Waleed Ammar. Scispacy: fast and robust models for biomedical natural language processing. *arXiv preprint arXiv:1902.07669*, 2019.

[7] Anusri Pampari, Preethi Raghavan, Jennifer Liang, and Jian Peng. emrqa: A large corpus for question answering on electronic medical records. *arXiv preprint arXiv:1809.00732*, 2018.