TreatME: A semantic patient-similarity-based collaborative filtering treatment recommendation system

Shuai Xu
Ming Hsieh Department of Electrical and Computer Engineering
University of Southern California
Los Angeles, CA 90007
sxu75374@usc.edu

***Abstract*—**
Nowadays, medication errors happen frequently, leading to severe consequences. 42% of the medication errors are caused by the lack of medical knowledge of new drugs or experience with some diseases. In recent years, medical recommendation has played an important role in the related fields. In this paper, we constructed a semantic patient-similarity-based collaborative filtering treatment recommendation system, called TreatME. We used lexical databases ConceptNet and NLTK WordNet to help measure word semantic similarity and sentence semantic similarity by treating the patient's symptoms as a sentence including semantic and structural information. Then, we implemented the collaborative filtering approach on the recommendation system to recommend top-k treatments based on the top 3 similar patients. Our proposed model shows a higher precision@k, recall@k, and f1@k score compared with the trivial model, popularity-based baseline models, and Jaccard-similarity-based model. The performance of TreatME is promising to serve as a treatment recommendation system application in daily use based on a large dataset.

***Key Words*— Treatment Recommendation System, Collaborating Filtering, Natural Language Processing, Semantic Analysis, Semantic Similarity, lexical database, Fuzzy Search**

I. Introduction

Recommendation systems are widely used in e-commerce, social media, and web services [1]. Furthermore, in recent years, the Recommender system plays an important role in the medical area. Medication error happens frequently, which may lead to serious results. 42% of medication errors are caused by doctors who lack experience with some diseases and prescribe drugs based on their limited experiences [2]. Besides, the fast-developed drug database will lead to a problem: it's difficult for doctors to learn all those new drugs in a short time and remember all the effects of drugs, and then prescribe them accurately.

By exploring the knowledge and relationships in the patient record database, Treatment Recommendation System (TRS) may help doctors precisely prescribe drugs or suggest treatments for the patients by exploring the knowledge and relationships in the patient record database. This is a good way to help make healthcare decisions for both patients and professionals. TRS, on the professional's side, could help them find similar patients who have similar symptoms or specific conditions based on the diagnosis history records. It's easier for doctors to suggest treatments for new patients and identify drug prescriptions more accurately [1, 2, 3]. Also, on the patients' side, the TRS could dynamically provide some suggestions for users, like doing more exercise, stopping smoking, etc., in time by analyzing users' health status and monitoring their lifestyle [4]. Personalization will be another important aspect of the medical recommendation system, which could effectively recommend the right doctor with good patient feedback or the right hospital to save money and save time [5]. Furthermore, helping patients find healthcare personalized precisely trustable therapy is another way to use a recommender system [6, 7].

Related studies have shown there are several ways to build the recommendation system and refine the results

of the recommendation. Collaborative filtering [6, 8, 9], knowledge-based (content-based filtering) [10, 11], and hybrid filtering [11] are three common methods used in recommender systems [4, 12]. Some recommender systems are based on domain ontology and lexical databases [13, 14]. Also, researchers used different NLP word vectorization methods in the medical recommendation system to preprocess the text data, like word2vec [14] and TF-IDF vectorization. Then, researchers used different text mining similarity measurements to find similar patient symptoms. Word similarity and semantic sentence similarity [15] in related works include Mahala Nobis distance [16], cosine similarity [17], co-occurring word-based similarity [18], and TF-IDF sentence similarity [18].

In this paper, inspired by the semantic similarity measures based on the semantic net [19][20], we proposed a novel semantic patient-similarity-based collaborative filtering treatment recommendation system, called TreatME, based on lexical databases WordNet [21] and ConceptNet [22]. The application of this research is to build a patient-similarity-based recommendation system based on the semantic similarity of patients' symptoms and conditions. We are going to consider a sequence of symptoms as a sentence where symptoms are ordered by frequency or severity. Here the semantic similarity will include both word and sentence similarity. Then, to best capture similar symptoms or diseases, we use both lexical databases ConceptNet and WordNet to find synonyms as a fuzzy search method. If we type in some symptoms that do not exist in the dataset, a fuzzy search could increase the probability to find possible treatments. Most of the related works are based on the EHR (electronic health record). Our experiment is based on a real-world patient information dataset and the experiment shows that the patient similarity-based semantic net treatment recommendation system, we called TreatME, achieves better performances in treatments recommendation.

In conclusion, we make some contributions as follow:

- We build a semantic patient-similarity-based treatment recommendation system, inspired by the NLP semantic net [19][20]. TreatME will base on both lexical databases ConceptNet and WordNet and their corpus statistics. ConceptNet could find synonyms for more phrases than WordNet, but WordNet has better corpus statistics and higher efficiency than ConceptNet. ConceptNet and WordNet could complement each other to improve retrieval performance.
- We measure the patient similarity by semantic word similarity and semantic sentence similarity. The semantic net gives us an idea to consider a sequence of symptoms as a sentence but not just consider each one of the symptoms separately, because the symptoms are listed by the frequency or severity. Because symptoms in one 'sentence' will not duplicate, using corpus statistics semantic vector and word order similarity could better measure the sentence similarity in TreatME than using TF-IDF which depends on the frequency of word occurrence or word2vec.
- Rather than searching for exact symptoms within the dataset, we use a fuzzy search method to help better capture similar symptoms and conditions, such as colloquial terms. A fuzzy search could help cover more potential similar patients.

The rest of the paper is organized as follows. Section 2 reviews related work and introduces the difference between our method and existing methods. Section 3 provides data sources and explains our methodology. Section 4 evaluates our recommendation system and shows the results. In section 5, we discuss the limitations of our project and future work. In section 6, we give the conclusion of our work. Section 7 discusses future work.

II. BACKGROUND/RELATED WORK

The recommendation system in the healthcare domain is like the information query and filtering system in other areas. It performs an important role in the future medical area to help make accurate make healthcare decisions. There are three most common recommendation algorithms, which are Collaborating Filtering (CF), Content-based Filtering (CB), and Hybrid Filtering (CF+CB).

2.1 Recommendation System Framework

Gräßer et al. [6] created a therapy decision-making system based on CF to recommend the most potential effective therapy for patients with an exclusion criterion that could help exclude inappropriate recommendation results. However, their project simply uses vector representation to measure user similarity and lacks consideration of semantic similarity. Also, their dataset has high dimensionality and has only 21 different outputs.

Zhang et al. [9] designed a neighborhood-based collaborative filtering recommendation system to predict the drugs' potential side effects. Researchers create a sparse matrix to represent the drug-side effect association and predict possible side effects based on the sparse matrix. However, this project lacks consideration of the severity of side effects, where the same side effects in different orders may be caused by different drugs.

Han et al. [18] conducted research on both content-based filtering and hybrid filtering recommendation system to recommend doctors for patients, based on patient-doctor interactions. CB could measure similarities between items based on users' preferences or ratings. CF could measure user similarities based on their preferences. However, both CB and Hybrid filtering (CF+CB) need

the information of users' ratings or preferences to measure the similarities and recommend items. CF will be more suitable for a clinic recommendation system, because, unlike drugs, patients usually don't have a preference or ratings for some of the treatments. There is more likely to find similar patients and suggest treatments based on those similar patients without requiring preference knowledge of treatments and patients.

2.2 Patient Similarity

Researchers used different word embedding methods to convert text data into numeric vectors. Common methods are word2vec [14] and TF-IDF vectorization [18]. However, word2vec has a big problem that could not handle out-of-vocabulary words. Although TF-IDF considers the term frequency in the sentence based on the corpus, both methods have a shortage - they ignore the order of the words in sentences, which lose some important semantic information.

Based on the embedded word vectors, different similarity measurements are used to find similar patients. Including both word similarity and sentence similarity, related works include using Euclidean distance, Mahala Nobis distance [16], cosine similarity [17], co-occurring word-based similarity [18], and tf-idf sentence similarity [18]. Only focusing on one measurement may lead to losing potential information in other aspects.

2.3 Our works

Because there not be too many textual multilabel treatment recommendation systems without ratings, we build the semantic patient-similar-based collaborative filtering (CF) treatments recommendation system. To compare with the related work, TreatME utilizes semantic net methods based on both lexical WordNet and ConceptNet and WordNet which could help measure the semantic similarity by synsets and edges. For the sentences in our dataset, we treat the symptoms of each patient as an ordered sentence because the symptoms are ordered by the frequency or severity which is an important piece of information that should be utilized. We first measure semantic word similarity. Then, we use corpus-based content-information weighted semantic vectors to calculate semantic sentence similarity and combine it with the word order similarity instead of using word2vec or tf-idf. We cascade multiple similarity measures together to measure the overall similarity. Then, we use a fuzzy search method based on lexical databases ConceptNet and WordNet to help better capture similar symptoms and conditions.

III. METHODOLOGY

3.1 Data Source

The dataset is from the website PatientLikesMe.com, which is web-scraped by GitHub user sooryaR [9]. The original dataset has 3350 patients' information with 8 features, including, UserID, age, gender, city, state, condition, symptom, and treatment. Name, City, State, and Condition are single-column text features, Gender is a categorical feature, Age is a numeric feature, Symptom is a multi-columns text feature, and the final output is Treatment, which is a multi-columns text feature. For the basic idea, in this project, we will only use five of all the features that relate to the diseases and treatments: Age, Gender, Condition, Symptom, and Treatment.

Table 1: dataset information

Feature	**Feature type**	**Description**
UserID	Text	Patient's ID
Age	Numeric	Patient's age
Gender	Categorical	Patient's gender
City	Text	Patient's location
State	Text	Patient's location
Condition	Text	Patient's condition
Symptom	Text	Patient's symptoms
Treatment (target)	Text	Patient's Treatments

3.2. Data Preprocessing

3.2.1 Deal with missing values

We first checked the missing values in the dataset and found that missing values exist in the column State and Treatment. Because Treatment is our ground truth, we only dropped those rows with Treatment=NaN.

3.2.2. Data Cleaning

Also, we found there exist some unexpected symbols in the Symptom and Treatment columns, such as 'redness)' and '"Itching'. We removed the unexpected right parenthesis symbol and quotation marks in the dataset. Also, we found some treatments are replaced by the symptoms in the dataset, like 'Pain",Fatigue' which are obviously not treatments but symptoms. After we remove rows with those keywords and fix the unexpected marks that occur in the features, there are 3,286 data points left in the dataset. There are 372 unique treatments, 240 unique conditions, and 147 unique symptoms in the dataset after data cleaning.

3.2.3. Mapping

Because both the Symptom and Treatment columns contain multiple words split by commas, we created a mapping to find the inner relationship between those features. We first set 1-to-N mappings: one condition

may have N different treatments, and one condition may have N different symptoms. Then, we created N-to-1 mappings: one symptom may have N different possible conditions and one treatment may use for N different conditions. We use nested dictionaries to create mappings. After obtaining the mapping relation, we could dig more into the dataset.

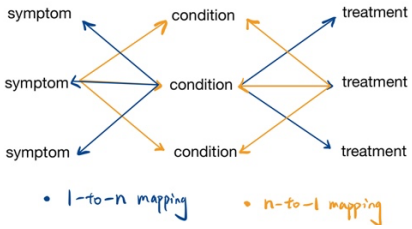

Figure 1: mapping

3.2.4. Train Test Split

We use the stratified train test split based on different conditions to obtain the train and test set with a ratio of 4:1. Finally, we have 2628 datapoints for the training set and 658 datapoints for the testing set.

3.3 Exploratory Data Analysis

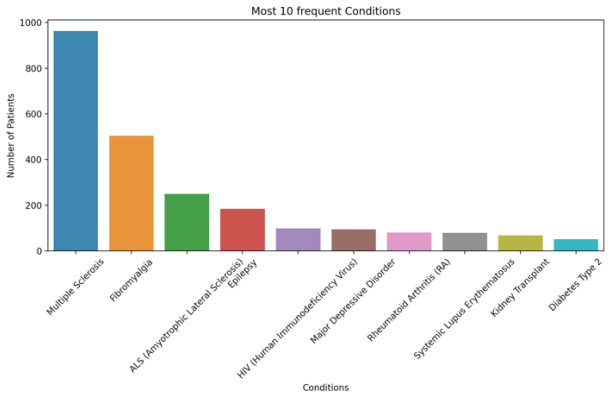

Figure 2: Top 10 common conditions people are suffering from

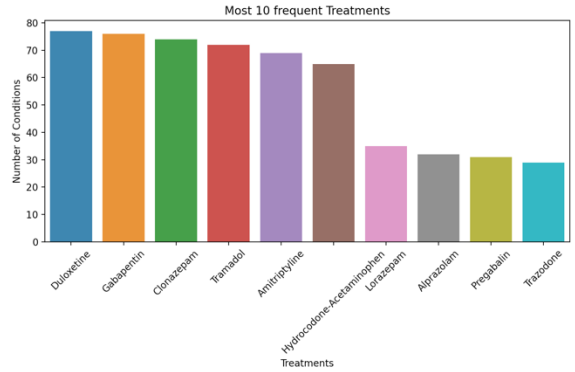

Figure 3: Top 10 common use treatments

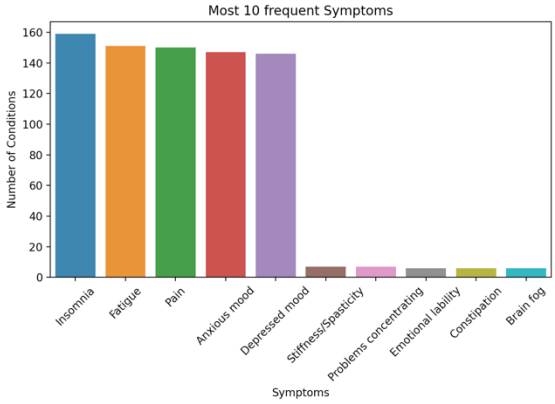

Figure 4: Top 10 frequent symptoms. Insomnia, Fatigue, Pain, Anxious mood, and Depressed mood are the most common symptoms

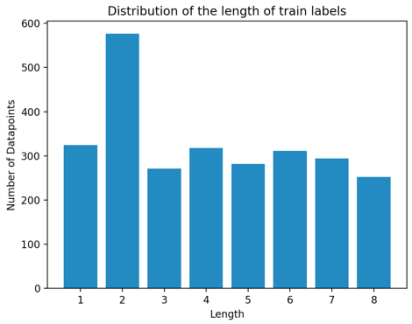

Figure 5: the distribution of the length of the training label

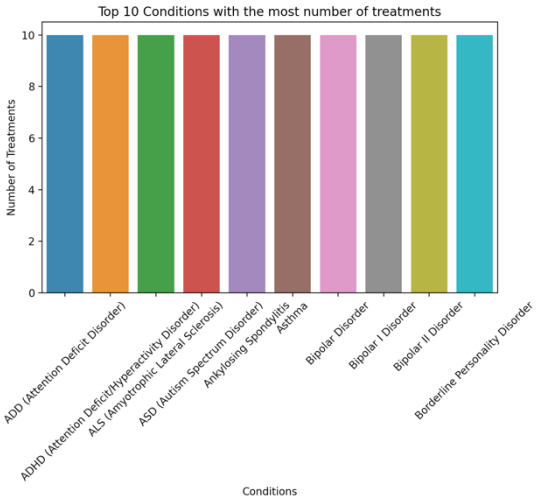

Figure 6: Top 10 conditions with the most number of treatments

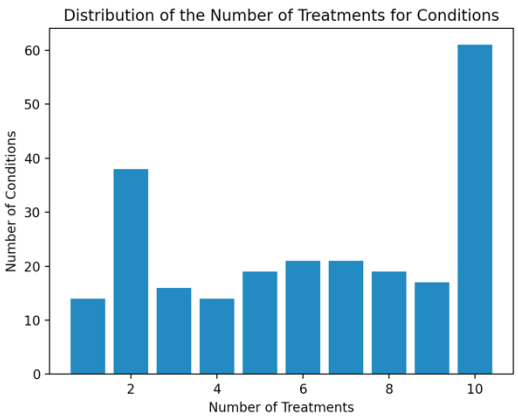

Figure 7: Distribution of Treatments for each condition

3.4 TreatME Framework

3.4.1 Collaborative filtering

Collaborative filtering (CF) is one of the most important recommendation algorithms used in the recommendation system. CF analyzes the pairwise similarity between users by using the knowledge of user history preferences and other users' preferences to recommend items – "similar users have similar tastes". In the healthcare domain, we use CF to find similar patients who are suffering from the same possible diseases by measuring the similarity of the patients' information and symptoms to recommend treatments for patients. CF will be more suitable for a clinic recommendation system, because, unlike drugs, patients usually don't have a preference or ratings for some of the treatments. Unlike content-based filtering, the collaborative filtering recommender system finds similar patients and suggests treatments based on those similar patients without requiring preference knowledge of treatments and patients [9, 28]. However, there is no Rating feature in our dataset and we could not simply treat this treatment recommender system like a normal RS, such as Movie RS. We can't use Matrix Factorization (MF) or Rating prediction which is based on user ratings to help recommend. Without the help of MF, NLP semantic or textual similarity techniques will help measure patient similarity.

In TreatME, we use semantic-similarity-based CF to find similar patients based on their age, gender, symptoms, and conditions. When recommending treatments, patients will get a list of recommended treatments. All those matches are based on similarities. In our recommendation system, we default to the Top 3 most similar patients and recommend N treatments based on their treatments. Figure 8 below shows the workflow of the TreatME system.

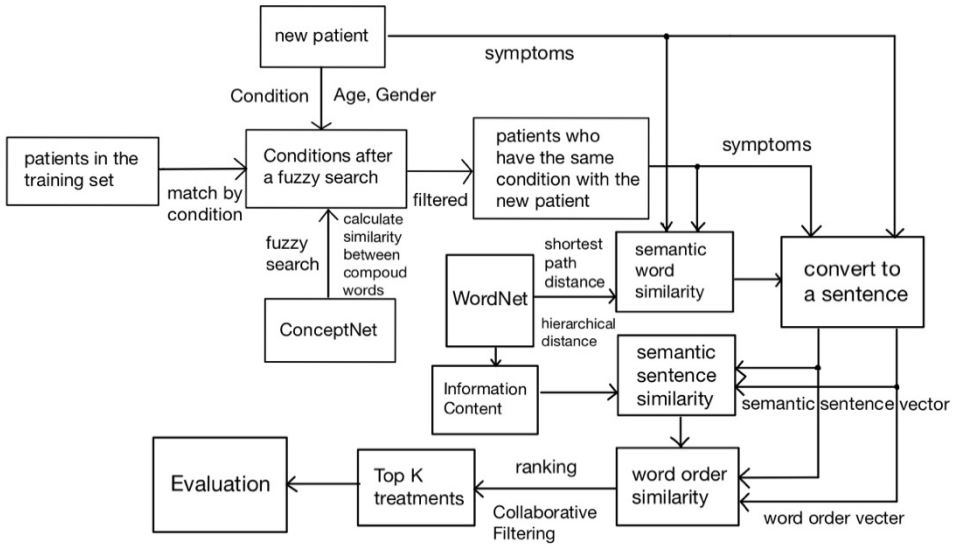

Figure 8: TreatME workflow

3.4.2 Patient Similarity

In our dataset, both conditions and symptoms for each patient are text data. Before we use NLP techniques to measure the patient similarity, we first use gender to filter the results. Then the difference in age between two patients should be no more than 10. Except for age and gender, each patient has one condition and several symptoms, and each condition and symptom could be one word or a phrase that is more than one word. In our paper, we consider those symptoms as one sentence, because patients' symptoms are ordered by frequency or severity which is a piece of important information. We

split the symptoms by the comma and join them into one sentence. Different orders of the same symptoms will potentially lead to a totally different diagnosis. Thus, the similarity measures could be split into two main parts: word similarity and sentence similarity.

3.4.2.1 Lexical Database and Corpus Statistics

Inspired by the sentence similarity that could be calculated based on the semantic net and corpus statistics workflow [19][20], we choose both lexical databases ConceptNet [22] and WordNet [21] to measure the word semantic similarity by the statistics of the hierarchical semantic knowledge base, like the distance between word synsets and edges connections. We use ConceptNet API to query the similarity between compound words in the matching part and find synonyms in the fuzzy search part. However, there is no feature in ConceptNet to get vertical hierarchical distance and horizontal distance between word synsets but much easier in WordNet. Thus, we use WordNet as the infrastructure for calculating corpus-based semantic sentence similarity. After we tokenize the sentence into tokens, each token could be utilized by WordNet to get the semantic word similarity for the further use of the semantic sentence similarity.

3.4.2.2 Word Semantic Similarity

From the article [19, 20], we could know the word semantic similarity contains two parts: horizontal path length and vertical hierarchical distance. For path length distance between synsets, I use NLTK WordNet shortest_path_distance(). For hierarchical distance between synsets, it's based on the Least Common Subsumer of the hypernyms. Hierarchical is very useful when word pairs have the same shortest path lengths [20]. The formula of hierarchical distance shows below:

$$D(h) = \frac{e^{\beta h} - e^{-\beta h}}{e^{\beta h} + e^{-\beta h}} \quad (1)$$

where α=0.2 and β=0.45, as the settings in the articles [19, 20]

There are three different similarity measures in the workflow. The first one is word semantic similarity. Assume the similarity between two words is based on both length and depth. We find the shortest path distance between the synsets of the two words for comparison based on path similarity. Then, find the scaling depth between two words based on the hierarchical structure of knowledge. This is the hierarchical distance.

3.4.2.3 Information Content

Before we talk about sentence similarity, we need to talk about the information content first. Charles T. Meadow, et al. [24] showed that the high-frequency words in the corpus contain less information compared with the low-frequency words in the corpus. Thus, the probability of the word occurring in the corpus could be considered as the information content of this word. The information content is negatively correlated with the word occurrence frequency. The formula shows below:

$$IC(\omega) = 1 - \frac{\log(n+1)}{\log(N+1)} \quad (2)$$

where ω is the word we are going to compute information content, n is the frequency of the word ω in the corpus, and N is the total number of words in the corpus.

3.4.2.4 Sentence Similarity

According to the articles [19, 20, 26, 29], sentence similarity consists of two parts: semantic sentence similarity and word order similarity.

Semantic sentence similarity. Semantic sentence similarity is measured by the cosine similarity between sentence semantic vectors. The semantic sentence vector is weighted by the words' information content. The length of the sentence vector is the length of the joint set of the two sentences we are going to compare. The sentence vector is initialed by the Bag of Words vector based on the joint set and common words in the sentence. If the joint word does not occur in the current sentence, then we use the most possible similar words in the sentence to fill it in. If the similarity of the words we find is lower than the threshold, we just remain zero in the sentence vector. Then, we weighted the sentence vector we get by the information content based on the formula below [19, 20]:

$$sv_i = \check{s}_i * IC(w_i) * IC(\widehat{w_i}) \quad (3)$$

where sv_i is one entry of the sentence semantic vector we are going to get the semantic vector, $\check{s}$ is the semantic word similarity, w_i is the word in the joint word set, $\widehat{w_i}$ is the associate word in the sentence. The semantic sentence similarity is calculated by the cosine similarity of the semantic vector:

$$S_{semantic} = \frac{sv_1 * sv_2}{||sv_1|| * ||sv_2||} \quad (4)$$

Word order similarity. In our data, like symptoms "Anxious mood, Depressed mood, Fatigue" and "Fatigue, Anxious mood, Depressed mood", obviously, they have the same Bag of Words vectors and semantic sentence vectors because both patients have the same three symptoms, but they are totally two different sentences with different meanings. The first sentence is from a patient who got Mild Depression, but the second

patient got Multiple Sclerosis. If we use the TF-IDF to vectorize those symptoms, we will get a similarity of 1, which means TF-IDF vectorization thought they were the same. Although TF-IDF considers the term frequency in the sentence based on the corpus [31], it ignores the order of the words in sentences [32], which means TF-IDF loses some important structure information. Word order similarity could help handle this case. We firstly vectorize the patient's symptoms to show the basic structure of the sentence, which is called a word order vector in the paper [20]. It assigns each unique word in the joint symptoms a number, which could represent the word (symptom) order. Then, measure the difference between those word order vectors to calculate the word order similarity. For the example above, the word order similarity of these two patients is 0.8325563283442158 to compare with the result of TF-IDF sentence similarity=1. The formula of word order similarity shows below:

$$S_{word_{order}} = 1 - \frac{||\boldsymbol{r_1} - \boldsymbol{r_2}||}{||\boldsymbol{r_1} + \boldsymbol{r_2}||} \quad (5)$$

Finally, we combine the sentence semantic similarity and word order similarity together based on the formula below to get the sentence similarity.

$$S(T_1, T_2) = \delta S_{semantic} + (1 - \delta) S_{word_{order}} \quad (6)$$

where S is the sentence similarity, δ is the weight of semantic similarity and word order similarity in the sentence similarity.

3.4.2.5 Compound Word Similarity

Although WordNet could easily measure the similarity of words and some open compound words (words spelled as two or more words), it's not good in some cases. First, if we treat a two-words term as two single words, the meaning of the terms may change, and add some unrelated non-medical words, and the tons of combinations of synsets for those two words will be way beyond what we need. Then, only parts of the medical terms can be found in WordNet, also some medical terms don't exist in WordNet, such as Atomoxetine, and Dystonia. In the fuzzy search part, some of the synonyms don't exist in WordNet but exist in ConceptNet, such as Myalgic Encephalomyelitis and Chronic Fatigue Syndrome. We only apply the ConceptNet word similarity to patients' condition but not both condition and symptoms, because we treat the symptoms as a sentence, but condition only has one medical compound word for each patient. Also, it will be time-consuming to query every word in the sentences and compare similarities from ConceptNet API if we want to apply ConceptNet to sentences. This is a trade-off. Thus, we use the ConceptNet to match those conditions, then we set a threshold=1 to filter the result, where similarity=2 means they are synonyms.

3.4.3 Fuzzy Search

A fuzzy search could help find possible matching patients who use the different colloquial terms or expert vocabularies [25] but have the same condition, such as Hypertension and High Blood Pressure. We use ConceptNet API to query synonyms of the conditions and add the synonyms to the patient's condition word set, then check if there are any of the conditions and synonyms hit the datapoints in the training set. If there exist new hits, we will add them to the checklist and compute the patient similarity again to find the new potential similar patients.

3.5 Baseline Models

In our recommendation system, we default to find the Top 3 most similar patients and recommend K treatments based on them. Based on this assumption, to compare the performance of TreatME on this dataset, we construct two baseline models:

- Trivial system: Randomly choose three patients from the training set and randomly recommend K treatments with replacement among their treatments without any prior knowledge.
- Baseline system: Randomly choose three patients from the training set among those patients who have the same condition as the patient we are going to recommend treatments for. Recommend the most frequently used K treatments based on those three patients.

Run both systems 10 times respectively and we take the average of the evaluation results as the final performance of those two systems.

Another method is based on a patient-similarity-based model using the ratio of common symptoms to measure the patient similarities. This is a patient-similarity-based method used to compare the performance of the model which implements semantic sentence similarity to measure patient similarities. This intersection-based similarity measure is called the Jaccard similarity coefficient [33]. The formula similarity shows below:

$$S_c(p_1, p_2) = \frac{|s_1 \cap s_2|}{|s_1 \cup s_2|} \quad (7)$$

where p_i is the patient and s_i is the symptoms of patient p_i.

3.6 Performance Evaluation

Our system predicts a list of recommended treatments for each of the patients. We assume the ground truth

treatments are ranked by the importance because the first treatment of a list of the treatments will be more useful or urgent for the patient. Precision@K, Recall@K, and f1 score@K are the commonly used evaluation metrics in the recommendation system, multilabel classification problem, and information retrieval system [13, 14, 27]. Precision shows the percent of correct recommendations by the number of recommendation results. Precision measures the system's ability to reject any nonrelevant data in the dataset. Recall measures the ability to find all relevant results [30]. Precison@K for top K recommendation is

$$\text{Precison@K} = \frac{|y \cap \hat{y}|}{|\hat{y}|} \quad (8)$$

Recall@k for top k recommendation is:

$$\text{Recall@K} = \frac{|y \cap \hat{y}|}{|y|} \quad (9)$$

F1 score for top k recommendation is:

$$\text{f1} - \text{score@K} = \frac{2 * Precision@k * Recall@k}{Precision@k + Recall@k} \quad (10)$$

where y is the ground truth (lists of treatments) and $\hat{y}$ is the prediction (lists of treatments).

IV. RESULTS

4.1 Figures and Tables

Table 2-4 shows the model performance. We compare the performance of those models with K from 1 to 7. The results show below:

Table 2: Model Performance(precision@k)

	precision@K						
Top K	1	2	3	4	5	6	7
Trivial system	0.0839	0.0718	0.0709	0.0702	0.0655	0.0607	0.0594
Baseline system	0.3809	0.3789	0.3794	0.3776	0.3746	0.3769	0.3787
Jaccard similarity model	0.4342	0.4283	0.4313	0.4266	0.4145	0.4113	0.4160
TreatME	**0.4658**	**0.4522**	**0.4421**	**0.4394**	**0.4360**	**0.4367**	**0.4329**

Table 3: Model Performance (recall@k)

	recall@k						
Top K	1	2	3	4	5	6	7
Trivial system	0.0197	0.0336	0.0499	0.0659	0.0768	0.0854	0.0976
Baseline system	0.0745	0.1474	0.2189	0.2861	0.3486	0.4102	0.4684
Jaccard similarity model	0.0941	0.1854	0.2788	0.3658	0.4392	0.5148	0.5907
TreatME	**0.0995**	**0.1904**	**0.2791**	**0.3668**	**0.4496**	**0.5266**	**0.5946**

Table 4: Model Performance (f1-score@k)

	f1-score@k						
Top K	1	2	3	4	5	6	7
Trivial system	0.0319	0.0459	0.0585	0.0679	0.0706	0.0709	0.0739
Baseline system	0.1247	0.2122	0.2776	0.3255	0.3612	0.3929	0.4188
Jaccard similarity model	0.1547	0.2588	0.3387	0.3939	0.4265	0.4573	0.4882
TreatME	**0.1639**	**0.2679**	**0.3422**	**0.3998**	**0.4427**	**0.4762**	**0.5011**

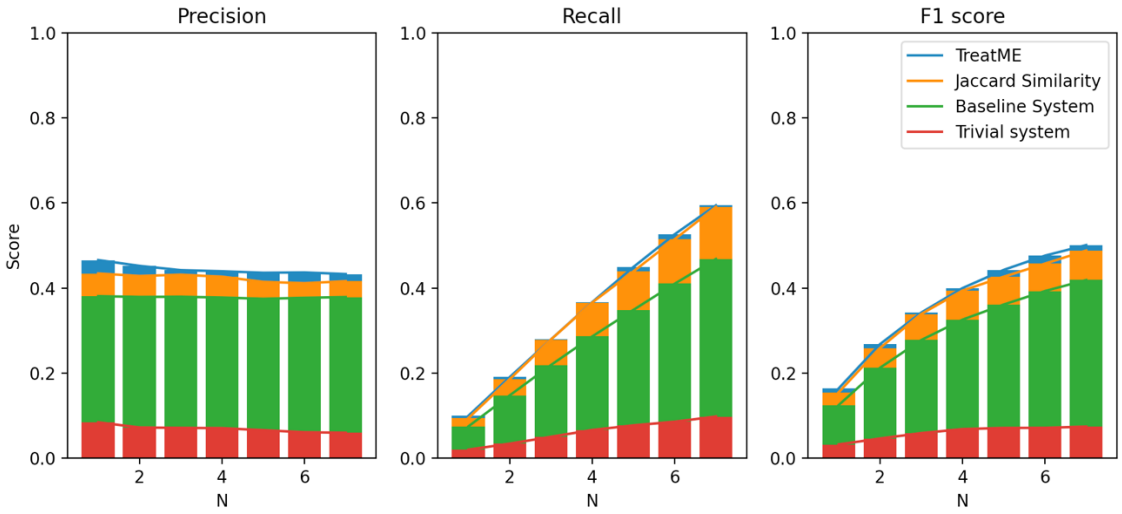

Figure 9: model performance comparison

4.2 Analysis

From Table 2-4, we could see that our model based on semantic sentence similarity outperforms the Jaccard similarity model and two baseline models. The overall performance is ordered by the semantic patient-similarity-based model (TreatME) > Jaccard similarity model > baseline system > trivial system.

TreatME system gives us the highest precision when we recommend the Top 1 treatment for patients. Table 2 shows that the overall precision of TreatME is about 1.1~3.1% better than the Jaccard similarity model, about 6~7 % better than the baseline model, and about 37~38% better than the trivial system. Table 3 shows that the difference between the recall of the TreatME system and the recall of the Jaccard similarity model is small. The range of difference between those two models is about 0.3% to 1.2%. What's more, based on different N, the recall of TreatME is about 2.5%~13% better than the baseline model, and about 8~50% better than the trivial system. From Table 4 we could find that the difference between the f1 score of the TreatME system and the Jaccard model is from about 0.4% to 1.9%. Based on different N, the f1 score of TreatME is about 4%~8.3% better than the baseline model, and about 13~42.7% better than the trivial system.

In Figure 9, we could find the precision curve has the tendency to go down but not changed much when N becomes larger. However, the precision of the baseline model is almost a horizontal line. This is because we randomly choose three patients who have the same condition as the patient in the test data and the system recommends the most frequently used treatments for this condition. Then, we generalize the baseline model by running the system 10 times and taking the average of the results as the final performance of the baseline system. This will cause the prediction of the baseline to follow the distribution of those most frequently used treatments and the distribution of the conditions in our dataset.

The precision curves of the other three models all have the tendency to go down. This is because when the N becomes larger, the cover rate of the recommended items will grow higher, and the recommendation system will recommend more unnecessary items. Because the recommendation results are ranked by the importance of the treatments and frequency based on the top three similar patients. Thus, when N is small, the recommendation system could more likely recommend useful treatments based on the ranking, giving us a high precision score.

For recall, we could find all the curves tend to go up. The recall in our dataset is the proportion of relevant treatments found in the top K recommendations we make [34]. When N becomes larger, more relevant treatments will be covered and recall becomes higher.

However, we could find that the difference between the recall of the Jaccard similarity model and the TreatME system is very small. Here is the reason. From Figure 6, we could see that in our dataset, the greatest number of treatments for one condition is only 10. That means when we use the similarity measure to narrow down the search, the range of the treatments will be fixed based on the patient's condition. When we use different similarity measures to find similar patients, there will be minor changes in the recall score because the similar patients are all based on the same condition, and treatments are all based on those treatments for that certain condition. Thus, it's not easy to make improvements to the recall score.

In Figure 7, the f1 score shows a tendency to go up when N becomes larger. When N becomes larger, the f1 score tends to converge.

Based on all the analysis above, we could say our model improves the accuracy of matching similar patients and gives a better overall recommendation result.

V. DISCUSSION

For the reasons why we construct those reference systems: The trivial system randomly recommends treatments without any prior knowledge. Our model could perform better than the trivial system means our model does learn something from the data. To compare with the baseline model, better performance means our model does not simply use the patient condition to predict treatments but also considers other information. Furthermore, we use the intersection of two patients' symptoms to measure the patient similarities (Jaccard similarity model). Jaccard similarity model does not just utilize the conditions' information but considers the similarity between patients. To compare with the baseline, we could find that the Jaccard similarity measure reduces the uncertainty of the prediction and improves the model performance which means the similarity measure is an effective method. Thus, we further implement our model to compare with the Jaccard similarity model to compare the performance of different similarity measures.

For the results, from Figure 9, we could see that the precision curve shows a tendency to converge to the baseline model when N becomes larger. That is because when N is large, the recommendation system tends to recommend more new items, but the number of all kinds of items is fixed and finite. Assume we have enough items to recommend. When N approaches infinity, the recommendation system will recommend all items to the user, which will cause the precision tend to converge to a point. This point will follow the distribution of the conditions and the distribution of their treatments.

From Figure 2, we could see that four conditions dominate the distribution of the conditions, which makes the dataset totally imbalanced. However, according to Figure 6, each condition has no more than 10 treatments. This is the main reason why the baseline looks strong and hard to beat. Hundreds of patients who got Multiple Sclerosis are all focused on the 10 treatments in the dataset. Thus, it means there will not be too many combinations of treatments for those patients who got Multiple Sclerosis. The baseline which is based on the most frequently used treatments will easily cover a high proportion of the ground truth.

Also, we think that when N becomes larger than six, the result becomes unreliable according to Figures 5-7. Based on the distribution of the length of the ground truth in Figure 5, we could see that most of the patients have no more than five treatments. Nearly 2/3 of patients have less than six treatments. Although we combine the Top 3 similar patients' treatments, they have the same condition, which means the number of treatments is finite (1-10). According to Figure 7, we find that 50.83% of the conditions have less or equal to six treatments. Also, the patient-similarity-based model may fail to recommend if there are no similar patients, which will cause the length of the prediction to equal zero. Thus, we think N should be small or equal to six in our dataset to consider both cases. When N is greater than six and becomes larger and larger, although the f1 score is better, the model will overfit, and the result will be biased.

In this dataset, the number of datapoints is not enough to get better performance, since there are lots of conditions that only have several datapoints. There are several ways could implement our model better theoretically: We could find a more balanced dataset or a dataset with enough datapoints to use. Our dataset is totally imbalanced and doesn't have enough datapoints. Also, our semantic model only focuses on the patient similarity based on their condition and symptoms, and auxiliary uses the age and gender to filter the results. Thus, a dataset with more features, especially text features, could better exploit our model's full potential.

Also, the time complexity of our method is higher than the other three models because we compare the sentence similarity of each patient one by one with pruning. There is a tradeoff in the information retrieval system between time efficiency, recall, and precision [35]. Our model performs better than the reference models, but it's time-consuming.

VI. CONCLUSION

In this paper, we developed a semantic patient-similarity-based treatment recommendation system by measuring sentence similarity. Inspired by the semantic net and corpus statistics, we used two lexical databases, ConceptNet and NLTK WordNet, to help measure word semantic similarity and sentence semantic similarity by treating the patient's symptoms as a sentence including semantic information and structural information. Then, implemented the collaborative filtering approach on the recommendation system to recommend top k treatments. In the experiment, we compared the performance of our model with the performance of the other three reference models.

For the performance of the trivial system, popularity-based baseline model, Jaccard similarity-based model, and our model with Top K where K from 1 to 6:

Table 5: Conclusion

	precision@k	recall@k	f1-score@k

trivial system	From 0.0594 to 0.0839	From 0.0197 to 0.0976	From 0.0319 to 0.0739
popularity-based baseline model	From 0.3746 to 0.3809	From 0.0745 to 0.4684	From 0.1247 to 0.4188
Jaccard similarity-based model	From 0.4113 to 0.4342	From 0.0941 to 0.5907	From 0.1547 to 0.4882
TreatME	From 0.4329 to 0.4658	From 0.0995 to 0.5946	From 0.1639 to 0.5011

Based on those evaluation metrics, we could conclude that our model outperforms the baseline models.

VII. FUTURE WORK

This research is extensible in different aspects. In the future, we will extend this research with auxiliary information, such as health status. With the help of the data, we could utilize other health-related features to predict the potential conditions, then recommend treatments based on the potential conditions we predict. Also, with the axillary data, we could exploit the interdependencies of the treatments to extend our CF model to a hybrid model (content-based filtering + collaborative filtering) to further improve the performance of the recommendation system. Time complexity is another problem we need to solve in the future.

REFERENCES

[1] Ko, H., Lee, S., Park, Y., & Choi, A. (2022, January 3). *A survey of recommendation systems: Recommendation models, techniques, and Application Fields*. MDPI. Retrieved March 23, 2022, from https://www.mdpi.com/2079-9292/11/1/141

[2] Y. Bao and X. Jiang, *An intelligent medicine recommender system framework,* 2016 IEEE 11th Conference on Industrial Electronics and Applications (ICIEA), 2016, pp. 1383-1388, doi: 10.1109/ICIEA.2016.7603801.

[3] Sahoo, Abhaya Kumar, et al. "DeepReco: Deep Learning Based Health Recommender System Using Collaborative Filtering." *Computation*, vol. 7, no. 2, 22 May 2019, p. 25, 10.3390/computation7020025. Accessed 8 Nov. 2019.

[4] Ali, Syed Imran, et al. "A Hybrid Framework for a Comprehensive Physical Activity and Diet Recommendation System." *Khu.elsevierpure.com*, Springer Verlag, 2018, khu.elsevierpure.com/en/publications/ahybrid-framework-for-a-comprehensive-physical-activity-and-diet-2. Accessed 24 Mar. 2022.

[5] Khoie, Mohammad, et al. "A Hospital Recommendation System Based on Patient Satisfaction Survey." *Applied Sciences*, vol. 7, no. 10, 21 Sept. 2017, p. 966, 10.3390/app7100966.

[6] Gräßer, Felix & Beckert, Stefanie & Küster, Denise & Abraham, Susanne & Malberg, Hagen & Schmitt, Jochen & Zaunseder, Sebastian. (2017). Neighborhood-based Collaborative Filtering for Therapy Decision Support.

[7] Fernandez-Luque, Luis, et al. "Challenges and Opportunities of Using Recommender Systems for Personalized Health Education." Medical Informatics in a United and Healthy Europe, 2009, pp.903–907, ebooks.iospress.nl/volumearticle/12798, 10.3233/978-1-60750-044-5-903. Accessed 23 Apr. 2022.

[8] Han, Qiwei & Martínez de Rituerto de Troya, Íñigo & Ji, Mengxin & Gaur, Manas & Zejnilovic, Leid. (2018). A Collaborative Filtering Recommender System in Primary Care: Towards a Trusting Patient-Doctor Relationship. 10.1109/ICHI.2018.00062.

[9] Zhang, W., Zou, H., Luo, L., Liu, Q., Wu, W., & Xiao, W. (2016). Predicting potential side effects of drugs by recommender methods and ensemble learning. Neurocomputing, 173(P3), 979–987.

[10] Han, Q., Ji, M., Martínez de Rituerto de Troya, I., Gaur, M., & Zejnilovic, L. (2018). A hybrid recommender system for patient-doctor matchmaking in primary care. In The 5th IEEE international conference on data science and advanced analytics (DSAA), (pp. 1–10).

[11] Doulaverakis, C., Nikolaidis, G., Kleontas, A., & Kompatsiaris, I. (2012). Galenowl: Ontology-based drug recommendations discovery. Journal of biomedical semantics, 3, 14.

[12] Adomavicius, G., and A. Tuzhilin. "Toward the next Generation of Recommender Systems: A Survey of the State-of-The-Art and Possible Extensions." IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 6, June 2005, pp. 734–749, 10.1109/tkde.2005.99. Accessed 25 May 2019.

[13] Doulaverakis, C., Nikolaidis, G., Kleontas, A., & Kompatsiaris, I. (2012). Galenowl: Ontology-based drug recommendations discovery. Journal of biomedical semantics, 3, 14.

[14] Jia, Zheng, et al. "A Patient-Similarity-Based Model for Diagnostic Prediction." *International Journal of Medical Informatics*, vol. 135, 1 Mar. 2020, p. 104073, reader.elsevier.com/reader/sd/pii/S1386505619310925?token=34AD41A20F588E512AA0C530C08D555695C3F9216FA2410F1784165C62E4919FB17D447F2231FDCA20E9504FEE823508&originRegion=us-east-1&originCreation=20220324073354, 10.1016/j.ijmedinf.2019.104073. Accessed 24 Mar. 2022.

[15] Farouk, Mamdouh. "Measuring Sentences Similarity: A Survey." Indian Journal of Science and Technology, vol. 12, no. 25, 1 July 2019, pp. 1–11, arxiv.org/ftp/arxiv/papers/1910/1910.03940.pdf, 10.17485/ijst/2019/v12i25/143977.

[16] F. Vitali, S. Marini, D. Pala, A. Demartini, S. Montoli, A. Zambelli, R. Bellazzi, Patient similarity by joint matrix trifactorization to identify subgroups in acute myeloid leukemia, JAMIA Open 1 (2018) 75–86, https://doi.org/10.1093/jamiaopen/ooy008.

[17] Tashkandi, Araek, et al. "Efficient In-Database Patient Similarity Analysis for Personalized Medical Decision Support Systems." Big Data Research, vol. 13, Sept. 2018, pp. 52–64, 10.1016/j.bdr.2018.05.001. Accessed 19 Dec. 2019.

[18] Ju, Chunhua, and Shuangzhu Zhang. "Doctor Recommendation Model Based on Ontology Characteristics and Disease Text Mining Perspective." *BioMed Research International*, vol. 2021, 8 Aug. 2021, p. 7431199, www.ncbi.nlm.nih.gov/pmc/articles/PMC8379386/, 10.1155/2021/7431199. Accessed 24 Mar. 2022.

[19] Li, Y., et al. "Sentence Similarity Based on Semantic Nets and Corpus Statistics." IEEE Transactions on Knowledge and Data Engineering, vol. 18, no. 8, Aug. 2006, pp. 1138–1150, 10.1109/tkde.2006.130. Accessed 12 Jan. 2020.

[20] Pawar, Atish, and Vijay Mago. "Calculating the Similarity between Words and Sentences Using a Lexical Database and Corpus Statistics." ArXiv:1802.05667 [Cs], 20 Feb. 2018, arxiv.org/abs/1802.05667. Accessed 23 Apr. 2022.

[21] G. A. Miller, "Wordnet: a lexical database for english," Communications of the ACM, vol. 38, no. 11, pp. 39–41, 1995.

[22] Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. "ConceptNet 5.5: An Open Multilingual Graph of General Knowledge." In proceedings of AAAI 31.

[23] "Treatment-Recommender/Crawler at Master · SooryaR/Treatment-Recommender." *GitHub*,

github.com/sooryaR/Treatment-Recommender/tree/master/Crawler. Accessed 24 Mar. 2022.

[24] C.T. Meadow, B.R. Boyce, and D.H. Kraft, Text Information Retrieval Systems, second ed. Academic Press, 2000.

[25] Martin Wiesner, and Daniel Pfeifer. "Health Recommender Systems: Concepts, Requirements, Technical Basics, and Challenges." International Journal of Environmental Research and Public Health, vol. 11, no. 3, 3 Mar. 2014, pp. 2580–2607, www.ncbi.nlm.nih.gov/pmc/articles/PMC3968965/, 10.3390/ijerph110302580.

[26] Mamdouh Farouk. "Measuring Sentences Similarity: A Survey." Indian Journal of Science and Technology, vol. 12, no. 25, 1 July 2019, pp. 1–11, arxiv.org/ftp/arxiv/papers/1910/1910.03940.pdf, 10.17485/ijst/2019/v12i25/143977. Accessed 29 Mar. 2020.

[27] Sanchez Bocanegra, Carlos Luis, et al. "HealthRecSys: A Semantic Content-Based Recommender System to Complement Health Videos." BMC Medical Informatics and Decision Making, vol. 17, no. 1, 15 May 2017, 10.1186/s12911-017-0431-7. Accessed 5 Aug. 2020.

[28] Roy, Abhijit. "Introduction to Recommender Systems- 1: Content-Based Filtering and Collaborative Filtering." Medium, 31 July 2020, towardsdatascience.com/introduction-to-recommender-systems-1-971bd274f421.

[29] Mamdouh Farouk. "Measuring Text Similarity Based on Structure and Word Embedding." Cognitive Systems Research, May 2020, 10.1016/j.cogsys.2020.04.002. Accessed 7 May 2020.

[30] Isinkaye, F.O., et al. "Recommendation Systems: Principles, Methods and Evaluation." *Egyptian Informatics Journal*, vol. 16, no. 3, Nov. 2015, pp. 261–273, www.sciencedirect.com/science/article/pii/S1110866515000341, 10.1016/j.eij.2015.06.005.

[31] Singh, Prasoon. "Fundamentals of Bag of Words and TF-IDF." Analytics Vidhya, 15 Feb. 2020, medium.com/analytics-vidhya/fundamentals-of-bag-of-words-and-tf-idf-9846d301ff22.

[32] Menon, Remya R. K., et al. "An Insight into the Relevance of Word Ordering for Text Data Analysis." 2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC), 2020, www.semanticscholar.org/paper/An-Insight-into-the-Relevance-of-Word-Ordering-for-Menon-dev/1586f23cbc0b49b9e1300abfa7156d1339bc10b7, 10.1109/ICCMC48092.2020.ICCMC-00040. Accessed 28 Apr. 2022.

[33] Stephanie. "Jaccard Index / Similarity Coefficient." Statistics How To, 3 Dec. 2016, www.statisticshowto.com/Jaccard-index/#:~:text=The%20Jaccard%20similarity%20index%20(sometimes.

[34] Maher Malaeb. "Recall and Precision at K for Recommender Systems." Medium, Medium, 13 Aug. 2017, medium.com/@m_n_malaeb/recall-and-precision-at-k-for-recommender-systems-618483226c54.

[35] Kobayashi, Mei, and Koichi Takeda. "Information Retrieval on the Web." ACM Computing Surveys, vol. 32, no. 2, June 2000, pp. 144–173, 10.1145/358923.358934.