ISSN 2395-8618

Graph Mining Healthcare Approach: Analysis and Recommendation

Hiba G. Fareed, Isam A. Alobaidi, Jennifer L. Leopold, Layth M. Almashhadani, Nathan W. Eloe

***Abstract*—Machine learning and computational intelligence have facilitated the development of recommendation systems for a broad range of domains. Such recommendations are based on contextual information that is explicitly provided or pervasively collected. Recommendation systems often improve decision-making or increase the efficacy of a task. An obvious application is a person's physical health where it is advantageous to increase the number of healthy cells in the body and destroy cancerous cells (wherein cancer is your opponent), we can learn how to predict positive outcomes for such scenarios. Herein we show how frequent and discriminative subgraph mining can be employed to analyze a collection of healthcare dataset cases and make recommendations about sequences of actions that should take, as well as should not take, be made to increase the chances of a patient's recovery in the near future. As proof of concept, we present the results of an experiment that utilizes our strategy for one particular healthcare dataset, MIMIC.**

***Index Terms*—Recommendation systems, graph mining, frequent subgraph mining, discriminative subgraph mining.**

I. Introduction

The quick expansion of information and development in communication technology has initiated a new era for researchers to develop e-health applications that play a major role in developing and improving healthcare services. Information growth requires competent and scalable techniques to generate useful results. Prediction systems were proposed as a computer-based intelligent technique to deal with the information problem and product overload.

These systems use knowledge discovery and statistical methods to recommend items to users. Medical data can have distinct types of characteristics and may contain various types of errors, such as missing or noisy data, which occur for a variety of reasons. For instance, a doctor may not request all appropriate tests while diagnosing a patient, some personal data may be neglected by users because of privacy matters, or values cannot be entered when the data is collected. Archiving accurate prediction is very challenging with such data, mainly due to the leak of explicit links between the actual state of the patients and recorded data.

Improving the accuracy of the prediction or recommendation system by using collaborative graph mining techniques was the primary goal of this work. This approach does not require any information other than the ICD-9-CM code to predict the medical conditions in order to assist physicians and patients. While clustering algorithms are commonly used for tackling similar problems, we chose the graph mining techniques due to the following issues associated with the clustering techniques: the need for data preparation, proximity measures, a method for handling outliers, and finally, reliance on a priori knowledge and user-defined parameters [1, 2, 3]. On the other hand, graph mining techniques are superior to clustering algorithms in terms of time complexity and are promising tools in data mining research.

In this study, we test the hypothesis that predictive analytics can be employed to examine a collection of patients' cases and make recommendations to increase the possibility of recovery. Using a database of patients' symptoms, we model each of those symptoms as a directed graph, and use frequent subgraph mining and discriminative subgraph mining, respectively, to look for patterns of symptoms that occurred in life cases; these form the basis of our recommendations for actions that a physician should take to keep the patient a live. Similarly, we look for patterns of symptoms that occurred in death cases; those become the basis of our recommendations for actions that a physician should not take to preserve a patient's life.

We test the accuracy of our two methods by partitioning our database of patients' cases into training and test datasets, and testing for the occurrence of true positives, true negatives, false positives, and false negatives. We also compare these two methods against each other, in terms of error rate of predictions. Scalability is a required feature of any method or system. The existence of this feature in our method made it possible to extend our proposed research to include the possibility of dealing with dynamic graphs, although our current scope of research deals with static graphs only. The use of parallel or distributed computing instead of sequential computing could be one of the proposed solutions for the future.

The organization of this paper is as follows. Section 2 provides a brief discussion of the main topics in this paper, including health recommendation systems and data mining techniques used in predictive analytics. The particular algorithms that we used for frequent subgraph mining and

Manuscript received on 18/09/2023, accepted for publication on 13/02/2024.

Hiba G. Fareed is with the Mathematics Department at Mustansiriyah University, Baghdad, Iraq (e-mail: hf_math@uomustansiriyah.edu.iq).

Isam A. Alobaidi is with the School of Computer Science and Information Systems at Northwest Missouri State University, Maryville, MO, USA (e-mail: ialobaidi@nwmissouri.edu).

Jennifer L. Leopold is with the Department of Computer Science at Missouri University of Science and Technology, Rolla, MO 65409, USA (e-mail: leopoldj@mst.edu).

Layth M. Almashhadani is with the Department of Computer Engineering, Al Farabi University College, Baghdad, Iraq (e-mail: layth.muhammad@alfarabiuc.edu.iq).

Nathan W. Eloe is with the School of Computer Science and Information Systems at Northwest Missouri State University, Maryville, MO, USA (e-mail: nathane@nwmissouri.edu).

ISSN 2395-8618

discriminative subgraph mining are explained in more depth in Section 3. A description of the MIMIC data that we used for testing our method is provided in Section 4. Our experimental method and results are discussed in Section 5. A summary of this research and consideration of future work is discussed in Section 6.

II. Background

In this section, the studies closest to the focus of our paper are reviewed, where researchers have developed tools for helping doctors and patients. In [4], the authors introduced a health recommendation system as a general predictive model to assess disease risks. The authors built a collaborative assessment and recommendation engine called CARE that relies on the collaborative filtering (CF) method for providing recommendations to patients by collecting preferences from users that have similar behaviors.

The utilized CF technique is derived from the vector similarity algorithm, which determines the similarity score based on the row vector. In another study [5], a hybrid recommendation system was proposed that combines CF with clustering on demographics of users with a weighted scheme. In this proposed system, item similarity and user clusters are computed offline, which makes the solution very scalable. In [6], CF was combined with techniques adopted from marketing domains and applied for the prediction of diseases.

The authors consider the inadequate medical history and locate other patients similar to a given person, who then vote on every disease the person has not yet had based on their own medical histories. The main limitations of the CF method are the cold start, sparsity, and scalability problems, which collectively affect the accuracy of the method.

The study presented in [7] has shown the application of the decision tree classification approach. Random forest was used to analyze and segment the patients' records in training data into distinguished and related groups of classes based on the observed diagnosis scales. Ensembles of decision trees, such as random forests, are very fast to train, but quite slow to create predictions once trained.

More accurate ensembles require more trees and thus operate more slowly. Another approach for disease prediction that combines clustering, Markov models, and association analysis techniques is proposed in [8]. The main weakness of Markov networks is their inability to represent induced and non-transitive dependencies; two independent variables will be directly connected by an edge merely because some other variable depends on both. As a result, many useful independencies go unrepresented in the network.

A. Data Mining Techniques Used in Predictive Analytics

Utilizing mathematical modeling, the field of predictive analytics examines past examples of life and death to determine the variables that lead to recovery outcomes and can be used to make predictions about future actions. It has been used widely in the financial and insurance sectors.

Here we briefly discuss some of the most common types of data mining methods used for predictive analytics.

Regression analysis: This method analyzes the relationship between a dependent variable and a set of independent variables. For healthcare data, the dependent variable would likely be the outcome of the cases (i.e., life or death), and the independent variables would be the various possible actions.

Rule induction: Rule induction methods such as association rule mining seek to find relationships between variables in the dataset [9]. By applying association rule mining on only the life cases, we could identify some actions that the physician did to save the patients. Similarly, by mining the death cases, we could find some actions common to losing patients.

Decision trees: Decision trees are most often used for classification and can be thought of as a graphical depiction of a rule; each branch of a decision tree can be thought of as a separate rule consisting of a conjunction of the attribute predicates of nodes along that branch [9]. One approach would be to construct decision trees from the life cases and death cases, respectively.

Clustering: Clustering is a way to categorize a collection of instances in order to look for patterns; groups are formed to maximize similarity between the instances within a group and to maximize dissimilarly between instances in different groups [9]. Health data are already clustered into two groups: life and death. For the purpose of analyzing successful (and unsuccessful) actions, we would likely attempt to form clusters of action sequences.

Neural networks: Neural networks are composed of a series of interconnected nodes that map a set of inputs into one or more outputs [9]. The interconnections between inputs (which, for the health data, could be actions in the case) could be determined based on an analysis of the patients' cases.

Most of the above methods would be computationally prohibitive, and would probably not yield useful results, for the MIMIC health data [25] unless we employed some type of data reduction mapping, which subsequently could result in loss of useful, specific information.

B. Subgraph Mining

Many problems can be modelled with graphs, wherein entities are represented as vertices and relationships between entities are represented as edges. When the relationship between two vertices has some semantic distinction of a predecessor and a successor, the edges are directed and hence the graph is considered directed. The MIMIC dataset can be modelled as a directed graph where each action (e.g., symptoms) is represented by a vertex and an edge represents two consecutive actions that were made. By necessity, each vertex also must be identified by which physician performed that action. The actions do not form a strictly linear sequence because an action can generate multiple actions; for example, the attending physician may decide to perform a quick surgical intervention while giving the patient a group of medications at the same time, each of which becomes a new vertex.

Subgraph mining is a technique used to discover a particular pattern in graphs. Two techniques will discuss here:

ISSN 2395-8618

— Frequent Subgraph Mining. Given a single (directed or undirected) graph, it can be useful to know which subgraphs occur at least n times where n is a user-specified threshold for frequency. Similarly, given a collection of graphs and a frequency threshold n, it may be important to know which subgraphs occur in at least n of those graphs. The process of answering this question is called frequent subgraph mining. Several methods for frequent subgraph mining were presented in [10,11,12,13]. Amongst many of the frequent subgraph mining algorithms that have been developed, computationally expensive extension/joining operations (to create larger candidate subgraphs from smaller frequent subgraphs) and false positive pruning (to reduce the search space) have been the biggest challenges that researchers have tried to address.

— Discriminative Subgraph Mining. Discriminative subgraph mining seeks to find a subgraph that appears in one collection of graphs but does not appear in another collection of graphs. This approach has been used to study several problems including identifying chemical functional groups that trigger side-effects in drugs [14], classifying proteins by amino acid sequence [15], and identifying bugs in software [16,17,18]. Various discriminative subgraph mining algorithms are given in [16,17,18,19,20], some of which are tailored for particular problems; due to space limitations, they are not discussed in detail here.

III. Methodology

In this section, we discuss the two graph mining methods that we utilized for the healthcare predictive recommendation system.

A. Frequent Subgraph Mining

Format and save your graphic images using a suitable graphics processing program that will allow you to create the images as PostScript (PS), Encapsulated PostScript (EPS), or Tagged Image File Format (TIFF), sizes them, and adjusts the resolution settings. If you created your source files in one of the following you will be able to submit the graphics without converting to a PS, EPS, or TIFF file: Microsoft Word, Microsoft PowerPoint, Microsoft Excel, or Portable Document Format (PDF).

1) Preliminaries

Let $\mathcal{G} = \{G_1, G_2, \dots, G_n\}$ be a set of linear directed graphs which represents the historical data. Each G_i represents a single symptom, such that $G_i = (V_i, E_i)$ where V_i represents a node labeled as a symptom code of a patient, while an edge in E_i represents two consecutive symptoms. A graph $T = (V_T, E_T)$ is a subgraph of $G_i = (V_i, E_i)$ iff $V_T \subseteq V_{G_i}, E_T \subseteq E_{G_i}$.

Definition 1. Let $T = (V_T, E_T)$ be a subgraph of a graph $G_i = (V_i, E_i)$. A subgraph isomorphism of T to G_i is an injective function $f: V_T \longrightarrow V_{G_i}$ satisfying $(f(u), f(v)) \in E_{G_i}$ for all edges $(u, v) \in E_T$. Intuitively, a subgraph isomorphism is a mapping from V_T to V_{G_i} such that each edge in E_{G_i} is mapped to a single edge in E_T and vice versa.

Problem 1. Given a set of graphs $\mathcal{G}$, the frequent subgraph isomorphism mining problem is defined as finding all subgraphs T in G such that $t_G(T) \geq \tau$, where $t_G(T)$ is the number of graphs in G that contain T and τ is the user-specified threshold.

Problem 2. Given a set of graphs $\mathcal{G}$ such that each G_i is divided into three phases G_{i1}, G_{i1}, G_{i3} and a frequent subgraph T, the frequent phase mining problem is defined as finding all subgraphs T in G_{ij} such that $t_{G_{ij}}(T) \geq \tau$, where τ is the user-specified threshold.

In our case, problem (2) counts the actual frequency (i.e., occurrences) of each subgraph provided that it is greater than or equal to τ. However, this may not be useful in various cases [13 and 21], while others necessitate the exact number of occurrences (like graph indexing in [22]). The choice of three for number of phases was an arbitrary decision influenced by board games such as chess that have traditionally been analyzed in terms of the moves made in the beginning, middle, and end of the game.

2) Electronic GraMi Algorithm

For the purpose of generating candidate subgraphs, a variety of frequent subgraph mining and subgraph extension algorithms have been developed, as discussed in previous work [12,22,23]. In particular, GraMi [23] is one of the most efficient methods and is the foundation for the work presented in this paper. The key ideas behind GraMi are briefly outlined here. Algorithm 1 is used to find a set of all frequent edges fEdges in the collection of graphs = $\{G_{i=1,\dots,n}\}$. All of these frequent edges have support greater than or equal to a user-specified threshold τ. Because of the anti-monotone property, only frequent edges are considered when finding the frequent subgraphs.

Algorithm 1 Frequent Subgraph Mining - FSM
Input $\mathcal{G} = \{G_{i=1,\dots,n}\}$ and frequency threshold τ
Output All $fSubgraphs\ S(G_i)$ with the support $\geq \tau$

```
1:  fSubgraphs ← ϕ
2:  Count = 0
3:      for each edge e_Gi do
4:          if e_Gi = e_Gi+1 then
5:              Count++
6:          end-if
7:          if Count ≥ τ then
8:              fEdges ← fEdges ∪ e_Gi
9:          end-if
10:     end-for
11:     for each e ∈ fEdges do
12:         fSubgraphs ←
        fSubgraphs ∪ SubE(e, G, τ, fEdges)
13:         Remove e from G and , fEdges
14:     end-for
15:   return fSubgraphs
```

Algorithm 2 is given each frequent edge to extend it to a new frequent subgraph. This is done by incorporating that edge with

ISSN 2395-8618

another subgraph. All extensions created in previous iterations are excluded by utilizing the DFScode canonical form that was introduced for gSpan [23]. The set Candidate in Algorithm 2 will include all the new subgraph extensions that had not been considered in prior iterations.

Algorithm 2 Subgraph Extension – $SubE$

Input $fSubgraphs\ S, fEdges$ and frequency threshold τ
Output All Sub_{new} with the support $\geq \tau$

$Sub_{new} \leftarrow \phi$
$Candidate \leftarrow \phi$
for each $e \in fEdges$ and $n \in fSubgraphs$ **do**
  if e fit to extend n **then**
    Generate a new subgraph $ExtS$
    if $ExtS$ exist in $\mathcal{G}$ and not generated before **then**
      $Candidate \leftarrow Candidate \cup ExtS$
    Else
      remove $ExtS$
    end-if
  end-if
end-for
for each $ExtS \in Candidate$ **do**
  if $ExtS$ count in $\mathcal{G} \geq \tau$ **then**
    $Sub_{new} \leftarrow Sub_{new} \cup SubE(ExtS, \mathcal{G}, \tau, fEdges)$
  end-if
End
return Sub_{new}

In subsequent steps, any new subgraph extension within the set Candidate that does not meet the support threshold τ requirement will be discarded. If any of those subgraphs had been extended, it would produce a new non-frequent subgraph according to the anti-monotonic property.

3) Using Frequent Subgraphs to Make Recommendations

In this section we discuss the algorithms that we utilized in order to mine the patient dataset for frequent subgraphs and build a recommendation system. The task of finding the number of occurrences for each subgraph was carried out using Algorithm 3.

Algorithm 3 Exact Subgraph Frequency

Input $\mathcal{G} = \{G_{i=1,\dots,n}\}, fSubgraphs\ S$ and frequency threshold τ
Output All the Exact Frequent Subgraph with their frequency

$Count = 0$
for $i = 1 \longrightarrow$ all graphs in $(fSubgraphs)$ **do**
  $frq = 0$
  for $j = 1 \longrightarrow$ all graphs in $(\mathcal{G})$ **do**
    if findnode $(G_j, fSubgraphs_i) \neq 0$ **do**
      $temp \leftarrow$ dfsearch $(G_j, fSubgraphs_i)$
      if $temp \geq size(fSubgraphs_i)$ & $isisomorphic(fSubgraphs_i, G_j)$ **do**
        $frq++$
      end-if
    end-if
  end-for
  if $frq \geq \tau$ **do**
    $count++$
    $ExactFSG(count) \leftarrow fSubgraphs_i$
  end-if
end-for
return $ExactFSG$

The mechanism for node-finding was used for matching the first node of a candidate subgraph with its occurrence in the original dataset. The objective of this process was to determine the starting point for conducting a depth-first search (DFSearch) to find all similar subgraphs in the recovery (or not recovery) graph collection. The expansion process will be done by adding new nodes that have met the threshold condition gradually. This process will contribute in maintaining the extracted subgraphs and avoiding recomputing everything from scratch. These results were stored temporarily in a temp set to compute their replication in the subsequent steps, and then the final result was placed within ExactFSG set.

B. Discriminative Subgraph Mining

The algorithm we employed for discriminative subgraph mining is similar to the approach taken in [17,18,26], but does not employ any heuristics specific to healthcare cases. Although we ran it sequentially, it easily lends itself to parallel or distributed processing.

Let R^+ and R^- represent two sets of (undirected or directed) graphs for which we want to find a discriminative subgraph; that is, we want to find a subgraph that appears in the graphs in R^- and does not appear in the graphs in R^+, or vice-versa. We shall refer to R^+ as the positive graphs and R^- as the negative graphs although this naming convention has no direct semantic correlation to the classification of the graphs in those respective sets (e.g., 'recovery' does not necessarily mean positive).

Algorithm 4 $FindDiscriminativeGraph\ (R^+, R^-, \alpha, \beta)$

R^+: set of positive graphs
R^-: set of negative graphs
α : percentage of graphs that discriminative subgraph need not be present in R^+ when relaxing conditions
β : percentage of graphs that discriminative subgraph need not be present in C^- when relaxing conditions

remove non-discriminative edges from graphs in R^+ and R^-;
$G = CreateDiscriminativeGraph(R^-, R^+)$;
if G is empty **then**
  $G = RelaxedCreateDiscriminativeGraph(R^-, R^+, |R^+| * \alpha)$;
  if G is empty **then**
    $G = CreateDiscriminativeGraph(R^+, R^-)$;
    if G is empty **then**
      $G = RelaxedCreateDiscriminativeGraph(R^+, R^-, |R^-| * \beta)$;
    end-if
  end-if
end-if
return G

The function $FindDiscriminativeGraph$ (Algorithm 4) first removes non-discriminative edges from the graphs in both sets; since such edges appear in the graphs in both sets, they cannot be used to differentiate the graphs in the those sets. $FindDiscriminativeGraph$ then calls $CreateDiscriminativeGraph$ (Algorithm 5) to try to find a subgraph that is common to all graphs in R^-, but not common to all the graphs in R^+. If we are unable to find such a graph, then the function $RelaxedCreateDiscriminativeGraph$ (Algorithm 6) is called, which relaxes the requirement that the

ISSN 2395-8618

subgraph we seek not be present in all of the R^+ graphs; instead the subgraph only has to not be present in $\alpha * |R^+|$ of the R^+ graphs, where α is a user-specified parameter (our default is $\alpha = 0.5$). $FindDiscriminativeGraph$ and $CreateDiscriminativeGraph$ use a function called Augment; this function takes a subgraph G and adds to it an edge (and possibly a node) such that the source vertex exists in G, and the edge (and destination node) exists in all graphs in subgraph collection S_1. In this way, a subgraph with an additional edge that exists in all elements of S_1 is created and considered by the algorithm.

Algorithm 5 $CreateDiscriminativeGraph\ (S_1, S_2)$

S_1: set of graphs
S_2: set of graphs
1: $FreqSG$ = queue of 1-edge subgraphs in S_1;
2: **while** $FreqSG$ is not empty **do**
3: $G = FreqSG$.dequeue ( );
4: **if** G is not in any graph in S_2 **then**
5: return (G);
6: **end-if**
7: NewGraphs = Augment (G);
8: **for** each graph G' in NewGraphs **do**
9: $FreqSG$.enqueue (G');
10: **end-for**
11: **end-while**
12: **return** (empty graph)

Algorithm 6 $RelaxedCreateDiscriminativeGraph\ (S_1, S_2, \gamma)$

S_1: set of graphs
S_2: set of graphs
γ : threshold for number of graphs discriminative subgraph must be present in
1: $FreqSG$ = queue of 1-edge subgraphs in S_1;
2: **while** $FreqSG$ is not empty **do**
3: $G = FreqSG$.dequeue ( );
4: **if** G is in $< \gamma$ graph in S_2 **then**
5: return (G);
6: **end-if**
7: NewGraphs = Augment (G);
8: **for** each graph G' in NewGraphs **do**
9: $FreqSG$.enqueue (G');
10: **end-for**
11: **end-while**
12: **return** (empty graph)

If we still fail to find a discriminative subgraph, then the difference likely does not involve edges that are in all graphs in R^- and not in graphs in R^+, but rather involves edges in the R^+ graphs that are not in the R^- graphs. Thus, we again call $CreateDiscriminativeGraph$, but reverse the order of the parameters (R^+ and R^-) from our previous call. If we still fail to find a discriminative subgraph, we again call $RelaxedCreateDiscriminativeGraph$ (Algorithm 6) and look for a subgraph that only has to not be present in $\beta * |R^-|$ of the R^- graphs, where β is a user-specified parameter (our default is $\beta = 0.5$).

It is possible that the resulting discriminative graph will be disconnected. Additionally, it could be the case that multiple subgraphs could qualify as a discriminative subgraph. The algorithm addresses both of these cases by returning the maximal discriminative subgraph; this result may be disconnected and will include all possible discriminative edges. It should be noted that it also is possible that our algorithm will not find any subgraph that meets the discriminative conditions. This could occur if the requirement that at least α (β) of the graphs in R^-(R^+) must have at least one edge in common has not been satisfied.

The computational complexity of the process is dependent upon the number of graphs in each collection and the number of edges in each graph. As specified in line 1 of $CreateDiscriminativeGraph$, we begin by examining each single edge from each graph in one of the graph collections. However, in lines 7-9 of that algorithm, we potentially build larger subgraphs that must be searched for; this is the subgraph isomorphism problem, which is NP-complete.

IV. Data Description

Medical Information Mart for Intensive Care (MIMIC) [25] is online medical data provided critical care data for over 40,000 patients admitted to intensive care units. MIMIC is made available largely through the work of researchers at the MIT Laboratory for Computational Physiology and collaborating research groups. In this study, a dataset of 46520 patients involving 2 cases, 30761 for life diagnoses ICD and 15759 for death diagnoses ICD, was obtained for diagnosed ICD patients. Each of these cases contained the sequence of diagnoses performed by each of two cases, with a designation of which patient is still alive. Each diagnosis in the dataset was encoded with 3-5 digits. Certain digits represented the patient ID, and other digits represented a counter (the number of times a particular patient presents with the same diagnosis).

V. Experimental Evaluation

In this section, we discuss the details of an experiment we conducted to test the hypothesis that predictive analytics, specifically frequent and discriminative subgraph mining, can be employed to examine a collection of patients' diagnoses and make recommendations as to what physician should do, and should not do, in order to increase the chances of making them recover in the near future.

A. Experimental Setup

We analyzed each patient-specific diagnosis group in one phase; the total number of diagnoses for each patient (by both the life diagnoses and death diagnoses) ranged from 6 to 358. For each phase analyzed, 80% of the data were used for training and the remaining 20% were used for testing with 10-fold cross-validation. A random number generator (www.random.org/lists/) was used to determine which patients were assigned to each partition (with no duplication). This process was repeated ten times for each phase in order to avoid any bias during the measure of error rate. Accuracy was used to evaluate the closeness of the measured value to the true value. Equation 1 is the mathematical formula of accuracy where TP is true positive, TN is true negative, FP is false positive, and FN is false negative:

ISSN 2395-8618

TABLE I
LIFE DATA OF FSM – MIMIC DATASET

Life cases Subgraph	Frequency	Classification
40391003 2851000 V4511001 V4582012	53	recover
V433003 412012 4241008 4280008	42	no action
82121001 82300001 E8217002 95892001	101	not recover
2851000 V4511001 E8217002 V433004	68	no action

TABLE II
DEATH DATA OF FSM – MIMIC DATASET

Death cases Subgraph	Frequency	Classification
2948022 78701001 2449001 V4501012	71	no action
E9352002 72141002 2148003 V5865003	39	not recover
V290003 V502005 V290004 V053012	59	not recover
4109002 V1007001 40390003 V290004 E8780001	63	no action

TABLE III
CROSS-VALIDATION TEST RESULTS OF FSM – MIMIC DATASET

Test No.	Phase 1 Avg. Error Rate	Phase 2 Avg. Error Rate	Phase 3 Avg. Error Rate
1	49.52%	32.16%	18.23%
2	52.93%	42.52%	13.63%
3	51.24%	40.61%	12.55%
4	55.13%	40.85%	14.32%
5	49.55%	39.12%	11.98%
6	50.94%	39.66%	14.44%
7	55.82%	39.88%	15.17%
8	48.59%	41.25%	11.82%
9	51.64%	38.22%	11.56%
10	49.66%	39.93%	13.54%
Avg.	**51.50%**	**39.42%**	**13.72%**

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}. \quad (1)$$

If a recommendation for what should be done to recover (subgraph) was found in one of the life graphs in the test partition, it was counted as a true positive (***TP***); if instead that recommendation (subgraph) was found in one of the death graphs in the test partition, it was counted as a false positive (***FP***). If a recommendation for what should not be done to avoid not recover (subgraph) was found in one of the death graphs in the test partition, it was counted as a true negative (***TN***); if instead that recommendation (subgraph) was found in one of the life graphs in the test partition, it was counted as a false negative (***FN***). The error rate was calculated as $\mathbf{1 - Accuracy}$, and was averaged over the ten iterations of the 10-fold cross validation. For each phase "1, 2, and 3" of the dataset cases, we used 10-fold cross validation. Ten partitions were created, each one contained 4652 cases; 10 iterations were run in those cross validations. By "cases" we mean both the life and death for that dataset cases.

B. Experimental Results

In this section, we present the results of analyzing the MIMIC dataset using both frequent subgraph mining and discriminative subgraph mining. The algorithms of frequent subgraph mining presented in Section 3.1 were (collectively) implemented in MATLAB and Java. The algorithms of discriminative subgraph mining presented in Section 3.2 were implemented in Python 3.7. A combination of Python programs and bash scripts were created for data file conversions and batch program executions. Our experiments were executed on an Intel(R) Core (TM) i7-6700 CPU@3.40GHz computer with 32GB memory.

1) FSM - Experimental Results

Tables 1 and 2, show some of the experimental results of frequent subgraph mining using a threshold of 10 for the life and death datasets consisting of 46520 patients diagnosed. The first and second columns show the diagnoses in the frequent subgraphs with their number of occurrences from the entire dataset, respectively. The third column in each table is a classification of the majority of that subgraph's actions; we classified that diagnosis actions as either recovered or not recovered (of an entity in the MIMIC dataset space). These results were obtained by performing 10-fold cross-validation, repeated ten times. Each time, for the 46520-case dataset, 41868 cases were selected randomly (without duplication) for training, and the remaining 4652 cases were used for testing.

For the MIMIC dataset, 809,356 frequent subgraphs were found that constituted "should do to recover" recommendations and 265,722 frequent subgraphs were found that represented "should not do to not recover" recommendations in phase 1. This number decreased in phase 2 to 472,535 frequent subgraphs were found that represented "should do" recommendations and 93,321 frequent subgraphs were found that characterized "should not do" recommendations. For the final phase of MIMIC, 108,271 frequent subgraphs were found that characterized "should do" recommendations; this was an 58.4% decrease from the number found in phase 1 and an 39.7% decrease from the number found in phase 2. In this phase, 23,545 frequent subgraphs were found that represented "should not do" recommendations; this was an 35.2% decrease from the number of such subgraphs found in phase 1 and an 25.3% decrease from the number found in phase 2.

The size ranged from two nodes with one edge to one hundred and seventy-seven nodes with one hundred and seventy-six edges in the MIMIC dataset. All of the two-node subgraphs were ignored because of the limited information they provide for the recommendation objective (i.e., only two diagnose) compared to larger subgraphs. Frequent subgraphs that were found in the life cases graphs indicate symptoms that are recommended for physician to accept, whereas frequent subgraphs that were found in the death cases graphs indicate symptoms that are recommended that physician should avoid.

The benefit of the counter attached to each symptom reflects the relative number of times had appeared that type of symptoms in that case. Characterizing the actions, such as recover or not recover, gives a general notion of the strategy the physician is employing in that sequence and would facilitate mapping one case's actions to another's.

Tables 3 show the average error rate for each of the cross-validation tests for each phase, as well as the average error rate over each phase's 10 tests for MIMIC, respectively.

ISSN 2395-8618

The resulting predictive accuracy was not good for frequent subgraph mining; in general, frequent subgraphs can have very low frequencies at times and high frequencies at other times. The collective recommendations (for actions that should be made and actions that should not be made) were accurate approximately 51.50%, 39.42%, and 13.72% of the time for phases 1, 2, and 3 of MIMIC, respectively. We attribute this increase in the error rate to the increase in the number of recover frequent subgraphs found in the death dataset and the not recover frequent subgraphs found in the life dataset.

2) DSM - Experimental Results

Tables 4 show the average error rate for each of the cross-validation tests for each phase, as well as the average error rate over each of the phase 10 tests for MIMIC, respectively. The resulting predictive accuracy was good, considering that, in general, discriminative subgraphs can have very low frequencies. The collective recommendations (for action that should be made and action that should not be made) had error rates of approximately 13.37%, 8.79%, and 1.53% of the time for phases 1, 2, and 3 of MIMIC, respectively. We attribute this decrease in the error rate to the decrease in the number of recover discriminative subgraphs found in the death dataset and the not recover discriminative subgraphs found in the life dataset.

For phase 1 of the MIMIC dataset, when testing all pairs of 2 life and 2 death graphs, 92,436 discriminative subgraphs were found that constituted "should do to recover" recommendations and 32,366 discriminative subgraphs were found that represented "should not do to not to recover" recommendations. The average size of the "should do" recommendation subgraphs was 26 edges; the smallest had 1 edge and the largest had 151 edges. The average size of the "should not do" recommendation subgraphs was 21 edges; the smallest had 1 edge and the largest had 133 edges.

The situation was not similar in phase 2 of MIMIC, where there were about 42% less than the respective numbers of subgraphs found in phase 1. When testing all pairs of 2 life and 2 death graphs, 43,636 discriminative subgraphs were found that represented "should do" recommendations and 16,441 discriminative subgraphs were found that characterized "should not do" recommendations. This is not surprising as the number (and order) of different actions that a physician could (and likely did) make increased at this point, thereby reducing the number of graphs that had edges in common and could meet the criteria of FindDiscriminativeGraph. The average size of the "should do" recommendation subgraphs was 19 edges, which was only slightly smaller than what had been found in phase 1; the smallest had 1 edge and the largest had 121 edges. The average size of the "should not do" recommendation subgraphs was 16 edges; the smallest had 1 edge and the largest had 93 edges.

In the final phase of MIMIC, 21,311 discriminative subgraphs were found that characterized "should do" recommendations; this was a 47.2% decrease from the number found in phase 1 and a 23% decrease from the number found in phase 2.

TABLE 4
CROSS-VALIDATION TEST RESULTS OF DSM – MIMIC DATASET

Test No.	Phase 1 Avg. Error Rate	Phase 2 Avg. Error Rate	Phase 3 Avg. Error Rate
1	13.32%	8.60%	0.92%
2	13.54%	8.08%	1.70%
3	12.95%	9.11%	0.80%
4	14.66%	9.20%	1.96%
5	13.22%	8.35%	1.45%
6	13.62%	9.20%	2.00%
7	12.10%	8.48%	1.46%
8	13.48%	9.52%	1.73%
9	14.35%	8.33%	1.63%
10	12.45%	9.11%	1.65%
Avg.	**13.37%**	**8.79%**	**1.53%**

In this phase, 9,3911 discriminative subgraphs were found that represented "should not do" recommendations; this was an 50.8% decrease from the number of such subgraphs found in phase 1 and a 29% decrease from the number found in phase 2.

The average size of the "should do" recommendation subgraphs was 18 edges; the smallest had 1 edge and the largest had 89 edges. The average size of the "should not do" recommendation subgraphs was 14 edges, which is close to the average size between what was seen for phases 1 and 2; the smallest had 1 edge and the largest had 86 edges.

Instead of looking at all the result subgraph (recommendations), the physician should be able to view only the top k "should" and "should not do" subgraphs, where k is a physician-specified parameter. For example, among the top ten frequently recommended "should do" subgraphs in phase 1 of MIMIC, 12 had 6 edges (i.e., 7 actions) and 18 contained 6-7 edges (i.e., 7-8 actions).

In contrast, 8 of the 10 most frequent "should not do" subgraphs contained 5-6 edges (i.e., 6-7 actions) and 2 contained only 1 edge (i.e., 2 actions). It should be noted that the "should" and "should not do" subgraphs can vary in the number of edges they contain; thus, we may not be able to provide as much information about what should not do as we can say about what should do (or vice versa).

The type of action can have an important role in characterizing a recommended subgraph (i.e., predominantly recover, not recover, or no action). In the MIMIC dataset, creation of territory files likely is considered a recover action. Another observation that can be made from discriminative subgraphs is a counter that is associated with both of these types of actions. For each patient, the counter for each type of action begins at 1 and is incremented by 1 each time that type of symptom appears. For example, edges (V4581023, 4019003, V4581024, V4581025, 53081001, V1051001) in phase 2 represent stability of the disease state in a certain period (actions beginning V4581) with counters 023, 024, and 025 (where the counter is initialized to 100), Indicated that the condition has stabilized after taking urgent action by taking a specific treatment. Their occurrence in a discriminative subgraph would indicate that it either is or is not advisable to take this action early.

ISSN 2395-8618

VI. Conclusion and Future work

The use of recommendation systems has become widespread in our society. In general, they examine historical data and try to predict what should be done in the future. Herein we have applied graph data mining techniques, frequent and discriminative subgraph mining, to healthcare system, MIMIC, to develop a system that can provide recommendations in order to improve a patient's chances of recovery. We modelled each case as a graph and found a collection of subgraphs that specified sequences of actions that physician should, and should not, make in each of three phases of the case. When testing datasets of both cases "life and death", experimental results of discriminative subgraph mining showed that the accuracy of our recommendations was high (an average of 93% accuracy for all three phases), and better than when using frequent subgraph mining.

Overall, our recommendations for our test were more informative in terms of what a physician should do at each of three phases in order to keep the patient a live and make him/her recover; however, we also were able to provide some information about what the physician should not do. Most importantly, this study has served as a proof of concept that the discriminative subgraph approach may be a promising strategy for not only healthcare predictive analytics, but also for other problem domains that involve direct and indirect resource generation and destruction. In the future we plan to establish a mapping between action types and assets so that a more generalized recommendation system can be developed. We also hope to explore ways to make the algorithms more efficient, perhaps applying some heuristics to reduce the search space that are inherent to the nature of healthcare data. Ultimately, we intend to abstract this strategy to other problem domains such as natural disasters such as earthquakes and hurricanes tracking and prediction systems using the same foundation of analyzing examples of survivance or not in order to make recommendations for future positive outcomes.

Acknowledgment

The authors would like to thank Mustansiriyah University[2] Baghdad, Iraq for its support in the present work. On the other hand, the authors thank Al Farabi University College for its support and our colleagues who provided insight and expertise that greatly assisted the research, although they may not agree with all of the interpretations/conclusions of this paper.

References

[1] A. Abugabah, A. Ahmad, and A. Abuqabbeh, "Data Mining in Health Care Sector: Literature Notes," in *Proceedings of the 2019 2nd International Conference on Computational Intelligence and Intelligent Systems*, pp. 63–68, 2019. DOI: 10.1145/3372422.3372451.

[2] K. H. Pine, C. Bossen, Y. Chen, G. Ellingsen, M. Grisot, M. Mazmanian, and N. H. Møller, "Data work in healthcare: Challenges for patients, clinicians, and administrators," in *Companion of the 2018 ACM Conference on Computer Supported Cooperative Work and Social Computing*, pp. 433–439, 2018. DOI: 10.1145/3272973.3273017.

[3] W. Fei, P. Zhang, and J. Dudley, "Healthcare data mining with matrix models," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 2137– 2138, 2016. DOI: 10.1145/2939672.2945387.

[4] N. V. Chawla, and D. A. Davis, "Bringing big data to personalized healthcare: a patient-centered framework," *Journal of General Internal Medicine*, vol. 28, no. 3, pp. 660–665, 2013. DOI: 10.1007/s11606-013-2455- 8.

[5] R. P. Song, B. Wang, G. M. Huang, Q. D. Liu, R. J. Hu, and R. S. Zhang, "A hybrid recommender algorithm based on an improved similarity method," *Applied Mechanics and Materials*, vol. 475, pp. 978–982, 2014. DOI: 10.4028/www.scientific.net/AMM.475-476.978.

[6] F. O. Isinkaye, Y. O. Folajimi, and B. A. Ojokoh, "Recommendation systems: Principles, methods and evaluation," *Egyptian Informatics Journal*, vol. 16, no. 3, pp. 261–273, 2015. DOI: 10.1016/j.eij.2015. 06.005.

[7] F. Folino and C. Pizzuti, "Combining Markov models and association analysis for disease prediction," in *International Conference on Information Technology in Bio-and Medical Informatics*, Springer, Berlin, Heidelberg, pp. 39–52, 2011.

[8] E. Alickovic and A. Suba, "Medical decision support system for diagnosis of heart arrhythmia using DWT and random forests classifier," *Journal of medical systems*, vol. 40, no. 4, pp. 1–12, 2016.

[9] A. Drachen, C. Thurau, J. Togelius, G. N. Yannakakis, and C. Bauckhag, "Game Data Mining," *Game Analytics: Maximizing the Value of Player Data*, London, UK, Springer London, pp. 205–253, 2013.

[10] J. Huan, W. Wang, J. Prins, and J. Yang, "SPIN: Mining Maximal Frequent Subgraphs from Graph Databases," In: *Proceedings of the 10th. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD `04*, Seattle, WA, USA, ACM, pp. 581–586, 2004. DOI: 10.1145/1014052.1014123.

[11] X. Yan, and J. Han, "CloseGraph: Mining Closed Frequent Graph Patterns," in *Proceedings of the 9th. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '03*, Washington DC., USA, ACM, pp. 286–295, 2003.

[12] J. Huan, W. Wang, and J. Prins, "Efficient Mining of Frequent Subgraphs in the Presence of Isomorphism," in *Proceedings of the 3rd. IEEE International Conference on Data Mining, ICDM '03*, pp. 549–552, 2003.

[13] M. Kuramochi and G. Karypis, "Finding Frequent Patterns in a Large Sparse Graph," in *Proceedings of the 2004 SIAM International Conference on Data Mining*, pp. 345–356, 2004.

[14] Z. Shao, Y. Hirayama, Y. Yamanishi, and H. Saigo, "Mining Discriminative Patterns from Graph Data with Multiple Labels and Its Application to Quantitative Structure-Activity (QSAR) Models," *Journal of Chemical Information Models*, vol. 55, no. 12, pp. 2519–2527, 2015.

[15] N. Jin, C. Young, and W. Wang, "Discriminative Subgraph Mining for Protein Classification," in *Computational Knowledge Discovery for Bioinformatics Research*, pp. 279–295, 2012.

[16] H. Cheng, D. Lo, Y. Zhou, X. Wang, and X. Yan, "Identifying Bug Signatures Using Discriminative Graph Mining," in *Proceedings of the 18th. International Symposium on Software Testing and Analysis ISSTA '09*, Chicago, IL USA, pp. 141–151, 2009.

[17] J. Leopold, N. Eloe, and P. Taylor, "BugHint: A Visual Debugger Based on Graph Mining," in *Proceedings of the 24th International Conference on Visualization and Visual Languages ICVVL '18*, San Francisco, CA, USA, pp. 109–118, 2018.

[18] J. Leopold, N. Eloe, J. Gould, and E. Willard, "A Visual Debugging Aid Based on Discriminative Graph Mining," *Journal of Visual Languages and Sentient Systems (VLSS)*, no. 4, pp.1–10, 2018.

[19] N. Jin and W. Wang, "LTS: Discriminative Subgraph Mining by Learning from Search History," in *Proceedings of IEEE 27th. International Conference on Data Engineering ICDE '11*, Hannover, Germany, pp. 207–218, 2011.

[20] X. Yan, H. Cheng, J. Han, and P. Yu, "Mining Significant Graph Patterns by Leap Search," in *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data SIGMOD '08*, Vancouver, BC, Canada, pp. 433–444, 2008.

[21] W.-T. Chu, and M.-H. Tsai, "Visual Pattern Discovery for Architecture Image Classification and Product Image Search," in *Proceedings of the 2nd. ACM International Conference on Multimedia Retrieval, ICMR '12*, Hong Kong, China, pp. 1–27, 2012.

[2] https://uomustansiriyah.edu.iq

ISSN 2395-8618

[22] X. Yan, P. S. Yu, and J. Han, "Graph Indexing: A Frequent Structure-based Approach," in *Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data, SIGMOD '04*, Paris, France, pp. 335–346, 2004.

[23] X. Yan and J. Han, "gSpan: Graph-based Substructure Pattern Mining," in *Proceedings of the 2002 IEEE International Conference on Data Mining, ICDM '02*, Maebashi City, Japan, pp. 721–724, 2002.

[24] M. Elseidy, E. Abdelhamid, S. Skiadopoulos, and P. Kalnis, "GraMi: Frequent Subgraph and Pattern Mining in a Single Large Graph," *Proc. VLDB Endowment*, vol. 7, no. 7, pp. 517–528, 2014.

[25] MIMIC Description, *Medical Information Mart for Intensive Care Description*" https://www.physionet.org/content/mimiciv/2.0/ Accessed: 2023-02-07.

[26] I. Alobaidi, J. Leopold, A. Allami, N. Eloe, and D. Tanksley, "Predictive analysis of real-time strategy games: A graph mining approach," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 11, no. 2, e1398, 2021.