IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 3, 2018, accepted September 17, 2018, date of publication September 24, 2018, date of current version October 17, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2871769

Automatic Diagnosis With Efficient Medical Case Searching Based on Evolving Graphs

XIAOLI WANG[1], YUAN WANG[2], CHUCHU GAO[1], KUNHUI LIN[1], AND YADI LI[3]

[1]School of Software, Xiamen University, Xiamen, China
[2]Department of Industrial Systems Engineering and Management, National University of Singapore, Singapore
[3]Department of Dermatology, Beijing Tongren Hospital, Beijing, China

Corresponding author: Yuan Wang (jessicawang36@gmail.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61702432, in part by the Fundamental Research Funds for the Central Universities of China under Grant 20720180070, and in part by the International Cooperation Projects of Fujian Province in China under Grant 2018I0016.

ABSTRACT The clinical data are often multimodal and consist of both structured data and unstructured data. The modeling of clinical data has become a very important and challenging problem in healthcare big data analytics. Most existing systems focus on only one type of data. In this paper, we propose a knowledge graph-based method to build the linkage between various types of multimodal data. First, we build a semantic-rich knowledge base using both medical dictionaries and practical clinical data collected from hospitals. Second, we propose a graph modeling method to bridge the gap between different types of data, and the multimodal clinical data of each patient are fused and modeled as one unified profile graph. To capture the temporal evolution of the patient's clinical case, the profile graph is represented as a sequence of evolving graphs. Third, we develop a lazy learning algorithm for automatic diagnosis based on graph similarity search. To evaluate our method, we conduct experimental studies on ICU patient diagnosis and Orthopaedics patient classification. The results show that our method could outperform the baseline algorithms. We also implement a real automatic diagnosis system for clinical use. The results obtained from the hospital demonstrate high precision.

INDEX TERMS Automatic diagnosis, multimodal medical data, graph similarity search, evolving graphs.

I. INTRODUCTION

Today, massive heterogeneous medical data have been collected from various healthcare organizations. Such data often contain rich knowledge such as diseases, drugs, and treatments. How to model the complex data for supporting effective analytic tasks on healthcare is a very important and challenging problem (e.g., [1]–[4]). Many researchers have focused on using information technologies for healthcare, and the disease diagnosis-aided system is one of the most important problems (e.g., [5]–[10]). To our best knowledge, there is no mature healthcare data analytic system in the medical industry.

Most existing work collected medical data as electronic health records (EHR), and adopted data-driven approaches to support a specific automate diagnosis task (e.g., [9]–[15]). Such systems can be classified into three categories as they employ different data modeling techniques: structured data models (e.g., [11], [12], [16]), semi-structured data models (e.g., [13], [14]), and graph data models (e.g., [9], [10], [15], [17]). Such systems have obtained some benefits such as more accurate diagnosis. However, they suffer from certain limitations for practical use as they have ignored the temporal property of clinical records.

Real clinical records of patients tend to be meaningful only when viewed against their temporal background [18]. Moreover, the temporal dimension is highly present in almost all critical medical scenarios. Thus, the comparison of patient cases on the basis of such data can offer valuable information for automatic diagnosis. To capture the temporal property, several existing work modeled medical data as time series data (e.g., [18]–[21]). They have employed several machine learning algorithms for predictive analytics. Although these proposals have reported accurate diagnosis, they may fail to capture both explicit and implicit relationships between various medical features. As a consequence, the overall analytical performance might be degraded.

To solve this problem, this paper models medical data as dynamic sequences of evolving graphs. The proposed data modeling approach could achieve two significant benefits.

- We build a personalized knowledge graph for each patient using the clinical data, by extracting entities and relationships between entities according to a semantic-rich medical knowledge base [10]. The profile graph of each patient could capture the explicit and implicit relationships between various medical features.
- To capture the temporal evolution of the patient's clinical case, we propose a novel data model as a sequence of evolving graphs. Each evolving graph represents a patient's visit in each time point. Each node in the graph represents a medical entity and each edge is the relationship between two entities.

We further investigate how to support effective prediction analysis. Most existing prediction methods on medical analysis often employ advanced machine learning algorithms such as Restricted Boltzmann Machine (RBM) (e.g., [22]), Principal Component Analysis (PCA) (e.g., [23]), and Gaussian Process (GP) (e.g., [21], [24]), to train models for diagnosis and prediction analysis. However, such methods often suffer from high computational cost for training models on the entire dataset. There are also some efforts on learning similar clinical cases for prediction analysis (e.g., [25]). These methods however suffer from low accuracy.

In this paper, we propose a new approach to employ the graph similarity search algorithm in graph sequence prediction. Our proposed method is different from existing work in that we take the implicit relationships in medical data into consideration, and capture the temporal evolution property of patients' condition. Two main technical obstacles must be overcome to make the graph sequence prediction scalable in practice. First, the expressivity of graph model relies heavily on the comprehensiveness of the medical knowledge base. We construct the personalized knowledge graph for each patient by using the historical clinical data. We extract the related entities and the relationships between entities according to the medical dictionary and knowledge graph. Second, to search the similar graphs, substantial searching effort at similarity computation time could lead to unacceptable latency. To address this problem, we define a novel similarity measure which attempts to evaluate the similarity between two sequence graphs. The novel measure can be efficiently computed in cubic time. To make graph search feasible in real-time applications, we develop an efficient graph similarity search algorithm. Specially, we design a novel three-level inverted index and propose a novel search strategy based on the TA algorithm [26]. In summary, our main contributions are listed as follows.

- We propose a novel modeling technique to represent complex medical data as sequence of evolving graphs. The objective is to take advantage of the rich expression power of graph models and effectively capture the temporal evolving property of medical data.
- We develop a novel lazy learning algorithm to support efficient automatic diagnosis based on evolving graph similarity search. To speed up the graph similarity search, a three-level inverted index is built, to support a better search strategy following a cascade framework.
- Two real datasets are collected to evaluate the effectiveness of our proposed approach. Both qualitative and quantitative evaluations demonstrate that our methods can indeed help doctors understand the patients' diseases and provide useful information for both ICU patient diagnosis by category and Orthopaedics patient classification. Furthermore, compared with the baselines, the proposed algorithms based on evolving graphs model have better performance.

The rest of the paper is organized as follows. Section II reviews the related literature. In Section III, we present an overview of our proposed method. Section IV describes our model for representing the complex medical data. We describe the basic profile graph similarity search algorithm in Section V, and propose an extended lazy learning approach to obtain more accurate diagnosis in Section VI. In Section VII, we summarize the experimental evaluation of our proposed method against the baseline algorithm. Finally, we conclude the paper and present future work in Section VIII.

II. RELATED WORK

A. MEDICAL DATA MODELING

Existing medial data modeling methods can be classified into four categories: structured data models, semi-structured data models, graph data models, and time-series data models.

Early medical data management systems mainly focus on storing medical record data in a structured way, such as EMR (e.g., [11], [16]). These systems are proposed to overcome the shortcomings of traditional handwriting records that are difficult to preserve and access. However, they often employ a simple relational model to store complex medical data, resulting in serious loss of medical structure information.

To capture the structure information, several semi-structured databases based on XML have been proposed [13], [14], [27]. These studies retain the original data structure using hierarchical models. However, many complicated information of the unstructured data are not completely recorded. As a consequence, such systems fail to support some complex analysis tasks on multimodal medical data.

Recent works employ the graph model to represent the complex medical data [9], [10], [15], [17], which can better represent the data relationships in the real world. The medical data analysis system based on the graph model consists of two main components: the construction of patient profile graph and the prediction analysis based on the graph model. Such systems have obtained some benefits such as more accurate diagnosis. However, they suffer from certain limitations for practical use as they have ignored the temporal property of clinical records.

Real clinical records tend to be meaningful only when viewed against their temporal background [18]. Therefore, several existing works modeled medical data as time series data (e.g., [18]–[21]). They have employed several advanced machine learning algorithms for predictive analysis. While these proposals have reported to have more accurate diagnosis, they fail to capture both explicit and implicit relationships between various medical features.

B. MEDICAL DATA STORAGE AND SEARCH

Different data searching techniques have been proposed based on the various data models. Early health data management systems used traditional relational databases for storage, and supported several simple data queries [11], [16]. For example, doctors can quickly query a patient's historical data using these systems. The aggregation functions of relational database can also be used to compute the statistics of patients. However, such systems cannot support complex analysis tasks.

To support multimodal data, several cross-domain retrieval techniques have been proposed [28]–[32]. These methods use the semantic information between multimodal data to build the linkage between various types of data and support cross-domain retrieval. These methods are usually applied to social media data of high correlation; while it is very difficult to adapt them for supporting medical data with ambiguous semantic relations. Several methods represent different modal data as different data types, and design a unified inverted index to support cross-domain search [33], [34]. Although these methods can support the query processing of different modal data, it cannot support complex analysis tasks.

With information explosion, healthcare big data analysis systems based on cloud computing are developed [35]–[38]. These medical cloud platforms solve extensive data storage requirements, and support simple data query and analysis. However, when dealing with complex data analysis requirements, they still suffer from lack of appropriate data fusion and modeling techniques. To our best knowledge, no mature healthcare data management and analytic system exists in the medical industry.

C. MEDICAL DATA ANALYTICS

Existing prediction methods generally follow into two groups: the eager learning approach and the lazy learning approach [39]. The eager learning approach often employs advanced machine learning algorithms such as Restricted Boltzmann Machine (RBM) (e.g., [22]), Principal Component Analysis (PCA) (e.g., [23]), and Gaussian Process (GP) (e.g., [21], [24]), to train models for diagnosis and prediction analysis. Many medical expert systems employed the eager learning approach, while such systems are rarely used in practice. Due to the limitations of machine learning algorithms in dealing with complex and dynamic environments, these systems often suffer from low accuracy of computer-aided diagnosis. The lazy learning approach typically is done by learning similar clinical cases based on search algorithms (e.g., [25]). These methods however suffer from low accuracy, as they often employ simple data models that results in serious loss of medical structure and semantic information.

TABLE 1. Notations.

Notations	Description
T	the unit tree
g	the profile graph
G	a sequence of evolving graphs
$T(g)$	the multiset of unit trees for the profile graph g
$\lambda(T_i, T_j)$	the distance between two unit trees T_i and T_j
$\lambda(g_i, g_j)$	the mapping distance between two profile graphs g_i and g_j
$\lambda(G^i, G^j)$	the alignment distance between two evolving graphs G^i and G^i
τ	the graph distance threshold

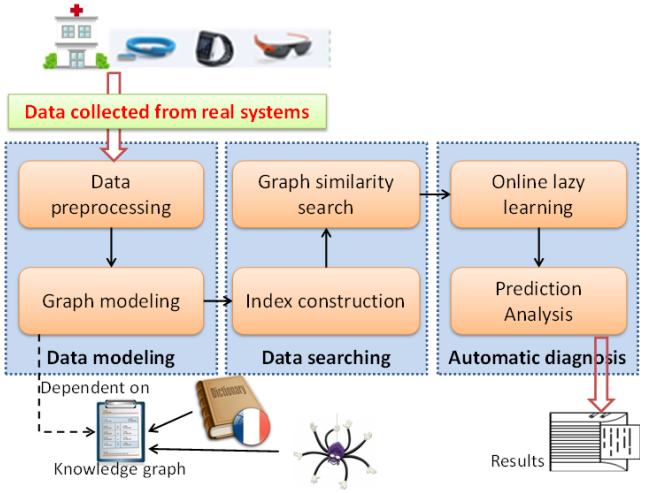

FIGURE 1. The system framework (Several icon photos in this figure are obtained from the Baidu search engine in https://image.baidu.com/.)

III. OVERVIEW

In this section, we give a formal description of the automatic diagnosis framework. Table 1 lists the common notations used in this paper.

As shown in Figure 1, our healthcare data analytics system contains three main components: data modeling, data searching and automatic diagnosis. We first build a semantic-rich knowledge base using both medical dictionaries and practical clinical data collected from hospitals. Based on the knowledge base, we build the graph model to bridge the gap between different types of data, and the multimodal clinical data of each patient are fused and modeled as one unified profile graph. To capture the temporal evolution of the patient's clinical case, the profile graph is represented as a sequence of evolving graphs. After that, we develop a lazy learning algorithm for automatic diagnosis based on graph similarity search. The details of the three components will be presented in the following sections.

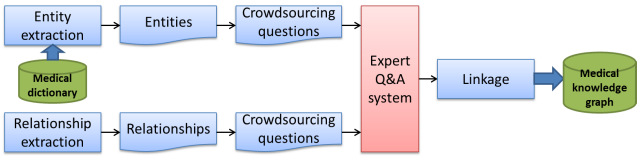

FIGURE 2. Construction of the medical knowledge graph.

IV. DATA MODELING

As shown in Figure 1, the data modeling component contains four steps: data collection, medical knowledge graph construction, data preprocessing, and data modeling.

A. DATA COLLECTION AND KNOWLEDGE GRAPH CONSTRUCTION

The raw data are collected from hospitals and wearable devices. We do data preprocessing to filter out the data of low quality, and then build a semantic-rich knowledge graph using the remaining data. Figure 2 shows the steps we employ to build the knowledge graph, and the detailed data preprocessing techniques and constructing algorithms can be seen in our previous work [40]. In general, the basic ontology of typical knowledge graph includes entity, category, attribute, and so on. For example, ''Subclass-of'' is often used to represent the affiliation relationship between an entity and a category (e.g., ''Andy Lau, Subclass-of, Singer''), and ''HasAttribute'' is used to represent the relationship between an entity and its attributes (e.g., ''Andy Lau, HasAttribute, male''). Our previous work employs both top-down and bottom-up approaches [41], [42] to build the ontology basis for knowledge graph [40]. Figure 3 presents the ontology basis of our constructed medical knowledge graph. To effectively modeling the medical data, we extract six types of entities such as Drug, Symptom, Disease, TestItem, Disease_Category, and Drug_Category. For each type of entities, we further extract representative attributes for it which may take significant roles in diagnosis. As shown in Figure 3, an entity named drug could have attributes of DOSE_VAL_RX and DOSE_UNIT_RX. We extract four other types of relationships as follows.

1) Diagnose: it is used to connect the relationship from the TestItem to the corresponding Disease.
2) Treat: it is used to connect the relationship from a Drug to the corresponding Disease.
3) Subcategory-of: it is used to connect the relationship from a Drug to a Drug_Category and from a Disease to a Disease_Category.
4) HasSymptom: it is used to connect the relationship from a Disease to a Symptom.

Note that existing medical dictionaries may lack domain specific relationships such as the relationship of ''Diagnose''. To solve this problem, as shown in Figure 2, our previous implemented *Expert Q&A system* [10] can support inferring the uncertain relationships by collecting answers from medical experts using crowdsourcing questions.

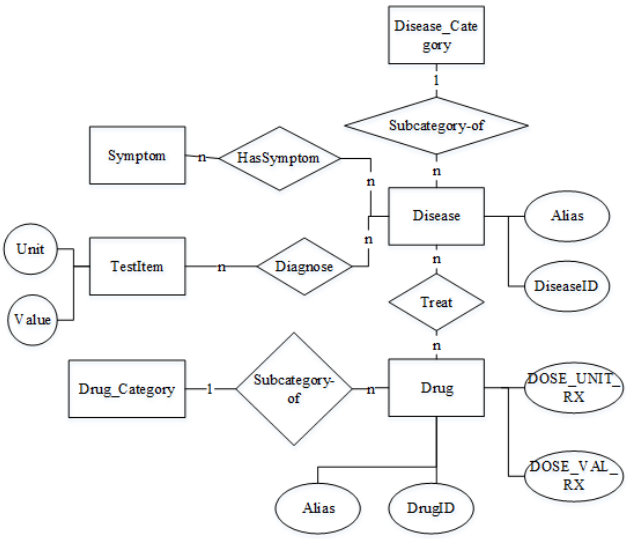

FIGURE 3. The ontology basis.

B. PATIENT PROFILE GRAPH

This paper employs the graph model to represent a patient's medical data, donated by *patient profile graph*. The patient profile graph is first introduced in paper [9] without a formal definition. The authors give a simple description that the graph contains vertices representing entities (e.g., diseases and medication) and edges representing relationships between entities (e.g., hasValue and diagnose). In this paper, we focus on the directed patient profile graph whose vertices and edges are labeled.

Definition 1 (Patient Profile Graph): A patient profile graph is defined as a 6-tuple $g = (V, E, \Sigma_V, \Sigma_E, l_V, l_E)$, V is a finite set of vertices, $E \in V \times V$ is a set of directed edges, Σ_V is a finite alphabet of vertex labels, Σ_E is a finite alphabet of edge labels, $l_V : V \to \Sigma_V$ is a labelling function assigning a label to a vertex, and $l_E : E \to \Sigma_E$ is a labelling function assigning a label to an edge.

We construct a patient's profile graph as a personalized knowledge graph for each patient. Our previous work proposed a disease diagnosis-aided system called ADDS [10] based on this data modeling technique. For each patient, we build a semantic-rich knowledge graph using the practical clinical data collected from hospitals and sensor data collected from wearable devices. Obviously, the expressivity of the patient profile graph relies heavily on the comprehensiveness of the medical knowledge base.

Based on the medical knowledge graph, we first collect patients' clinical data and sensor data. Then, entities are extracted and represented as vertices in the patient graph. If two entities have relationships in the knowledge graph, we add one directed edge between them. Figure 4 shows an example of the ontology for a patient profile graph. Suppose a clinical record indicates that doctors suggest to treat the disease with certain amount of drugs. Then, we use two attributes DOSE_VAL_RX and DOSE_UNIT_RX to represent the dose of the drug. In this case, two entities Disease and Drug

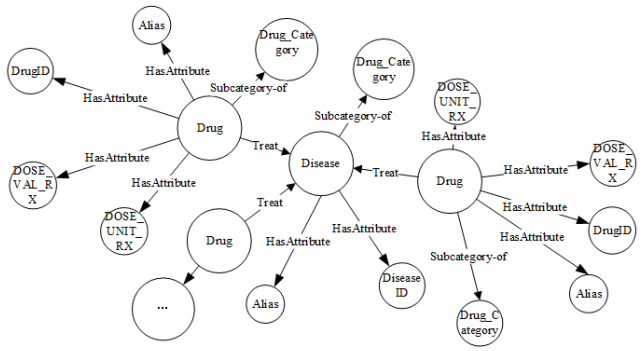

FIGURE 4. The ontology of a patient's profile graph.

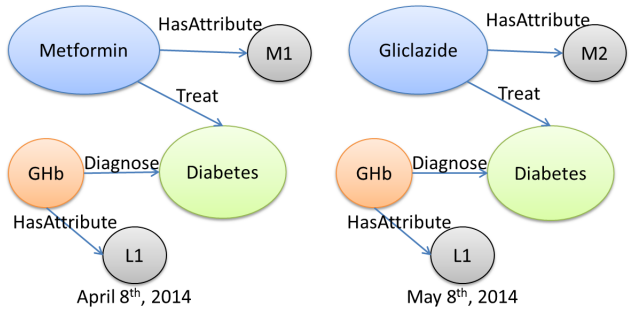

FIGURE 5. A patient's sequence of evolving graphs.

and two attributes DOSE_VAL_RX and DOSE_UNIT_RX are represented as four vertices; two directed edges are added from the Drug entity to two attributes representing the "HasAttribute" relationships; one directed edge is added from the Drug entity to the Disease entity representing the "Treat" relationship.

C. PATIENT's SEQUENCE OF EVOLVING GRAPHS

Clinical data generally capture patients' visits to hospitals, and tend to be meaningful only when viewed against their temporal background. In this paper, we abstract the clinical data as an evolving graph sequence where each graph represents a patient's visit at each time point.

Definition 2 (Evolving Graph Sequence): An evolving graph sequence of a patient i is defined as $G^i = \{g_1^i, g_2^i, \ldots, g_j^i, \ldots, g_{|G^i|}^i\}$. g_j^i is the profile graph at timestamp j. $|G^i|$ is the size of G^i representing the total number of patient profile graphs in G^i.

Figure 5 shows a real example of constructed evolving graph sequence for a patient. The patient has visited the hospital for two times respectively on April 8^{th}, 2014 and May 8^{th}, 2014. We build two profile graphs for these two visits. The first profile graph contains five vertices and four edges as follows.

- One vertex of Disease labeled as Diabetes.
- One vertex of Drug labeled as Metformin.
- One vertex of TestItem labeled as GHb.
- Two vertices of attributes, one of which is the attribute of GHb labeled as $L1$ and another is the attribute of Metformin labeled as $M1$.
- One directed edge from Metformin to the Disease labeled as Treat.
- One directed edge from GHb to the Disease labeled as Diagnose.
- Two directed edges from Metformin and GHb to their attributes both labeled as HasAttribute.

Noted that the clinical data collected from patient's one visit to hospital may contain massive information on multiple diseases. In this paper, we mainly focus on one type of disease, and eliminate the information of other diseases. Therefore, each evolving graph sequence contains a set of continuous patient profile graphs only related to one disease. In Figure 5, both two profile graphs are related to "Diabetes".

V. MEDICAL DATA SEARCHING

The comparison of patient cases on the basis of medical profile graph can offer valuable information in a variety of tasks, ranging from differential diagnosis to patients' recovery prediction. Based on the graph model, it is a straightforward idea to employ existing graph similarity measures [43], for evaluating the similarity between two medical cases. However, such approaches often have the common problem that they usually require high computational cost. For instance, computing the graph edit distance can be in NP-hard [44]. In this paper, we propose a novel similarity measure called *graph mapping distance* to evaluate the similarity between two patient profile graphs. The key idea is to transform a profile graph to a multiset of unit trees.

A. UNIT TREE

Definition 3 (Unit Tree): A unit tree is an attributed, single-level, rooted, directed tree which can be represented by a 5-tuple $T = (r, L, D, l_L, l_D)$, where r is the root vertex, L is the set of leaves, D is the set of edges from root vertex to the leaves, l_L is a labeling function assigning a label to a leaf vertex, and l_D is a labeling function assigning a label to an edge. Directed edges exist from the root vertex r to each vertex in L.

Given a patient profile graph g, for any vertex v_i in g which has out-degree, we can generate a corresponding unit tree T_i in the following way: $T_i = (v_i, L_i, D_i, l_{L_i}, l_{D_i})$ where $L_i = \{u|(v_i, u) \in E\}$. Thus, a graph can be mapped to a multiset of unit trees. We call this multiset the tree representation of the graph g, denoted by $T(g)$.

Lemma 1 [Unit Tree Distance (UTD)]: Given two unit trees T_1 and T_2, the distance between them is computed as

$$\lambda(T_1, T_2) = d(r_1, r_2) + d(D_1, D_2) + d(L_1, L_2)$$

where
$d(r_1, r_2) = 0$ if $l(r_1) = l(r_2)$, otherwise $d(r_1, r_2) = 1$

$$\begin{aligned} d(D_1, D_2) &= \big||D_1| - |D_2|\big| + M(D_1, D_2) \\ M(D_1, D_2) &= \max\{|\Psi_{D_1}|, |\Psi_{D_2}|\} - |\Psi_{D_1} \cap \Psi_{D_2}| \\ d(L_1, L_2) &= \big||L_1| - |L_2|\big| + M(L_1, L_2) \\ M(L_1, L_2) &= \max\{|\Psi_{L_1}|, |\Psi_{L_2}|\} - |\Psi_{L_1} \cap \Psi_{L_2}| \end{aligned}$$

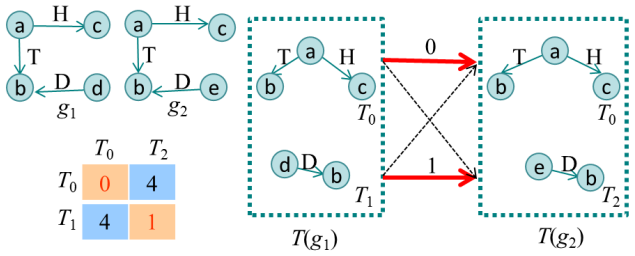

FIGURE 6. An example of computing the mapping distance between two profile graphs g_1 and g_2.

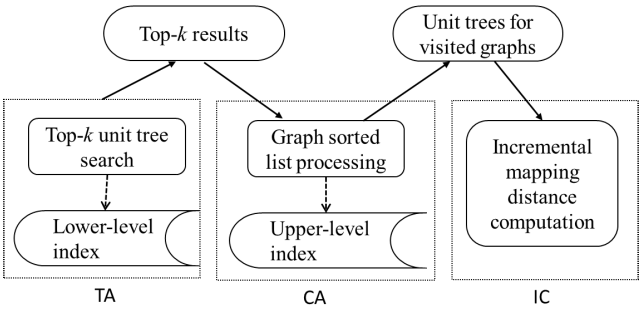

FIGURE 7. The framework of profile graph similarity search

Ψ_D is the multiset of edge labels in D and Ψ_L is the multiset of vertex labels in L.

B. GRAPH MAPPING DISTANCE

Based on the unit tree representation of graphs, a polynomial time computable distance is introduced in this subsection. That is, we define a mapping distance between two profile graphs based on their multisets of unit tree representations.

Definition 4 [Graph Mapping Distance (GMD)]: Given two graphs g_1 and g_2 with their multisets of unit trees $T(g_1)$ and $T(g_2)$ of the same cardinality, and a bijection $P : T(g_1) \rightarrow T(g_2)$, the mapping distance between g_1 and g_2 denoted by $\lambda(g_1, g_2)$ is computed as

$$\lambda(g_1, g_2) = \lambda\big(T(g_1), T(g_2)\big) = \min_{P} \sum_{T_i \in T(g_1)} \lambda(T_i, P(T_i)).$$

The computation of $\lambda(g_1, g_2)$ is equivalent to solving the assignment problem, which is one of the fundamental combinational optimization problems aiming at finding the minimum weight matching in a weighted bipartite graph. Given two multisets of unit trees $T(g_1)$ and $T(g_2)$ and the weight of the edge that connects one unit tree in $T(g_1)$ to another unit tree in $T(g_2)$ is the distance between the two unit trees. Figure 6 shows an example to illustrate the bipartite graph matching problem that must be solved in order to compute $\lambda(g_1, g_2)$. Given two graphs g_1 and g_2 on the left top of the figure, their corresponding multisets of unit trees are shown on the right hand side of the figure. In this case, we show the optimal matching between the unit trees as solid lines while other edges joining the unit trees are shown as dotted lines. To find the optimal matching, we construct a weighted matrix for each pair of unit trees from two graphs, and apply the Hungarian algorithm [45] to get the optimal solution in cubic time. The matrix is shown on the left bottom of the figure, and cells in red denote the optimal matching between $T(g_1)$ and $T(g_2)$, i.e., $\lambda(g_1, g_2) = 0 + 1 = 1$.

C. PROFILE GRAPH SIMILARITY SEARCH

In paper, Wang *et al.* [46] proposed a two-level inverted index to support efficient graph similarity searching algorithms. In this paper, we adapt it to support the proposed profile graph similarity searching, including range query and KNN query.

1) INDEX CONSTRUCTION

Given a set of profile graphs and their multisets of unit trees. In the upper-level index, unit trees are used to index all graphs using inverted lists. This index is made up of two main parts: an index for all distinct unit trees, and an inverted list below each unit tree. Each entry in the inverted lists contains the graph identity and the frequency of the corresponding unit tree. All lists are sorted in increasing order of the graph size.

In the lower-level index, each unit tree is further broken into multiple units (i.e., vertices and edges) and indexed in inverted lists. The index also contains two components: a label index in increasing order and inverted lists below labels recording the unit tree identities and the frequencies of corresponding labels. Entries in each list are first grouped based on the leaf size of $|\Psi_L|$ and then sorted in decreasing frequencies within each group.

2) PROFILE GRAPH SIMILARITY SEARCH

As shown in Figure 7, based on the two-level index, the profile graph similarity search includes two steps as follows.

- Step 1: Given a query g_q, we decompose it into a multiset of unit trees as $T(g_q)$. In the lower level, top-k similar unit trees to each unit tree of the query can be returned quickly by using the TA search algorithm.
- Step 2: In the upper level, graph pruning is done based on the top-k results from the lower level. The CA algorithm is employed to support continuous graph pruning. In this step, unit trees for each profile graph can be output with round-robin scan through the score sorted lists, and the mapping distance for each seen data graph with the query is computed incrementally. Obviously, the partial mapping distance can be naturally used as a lower bound for graph pruning.

Now we define a monotonic score function for the TA framework. From Lemma 1, the $\lambda(T_q, T_i)$ between a query unit tree T_q and any unit tree from profile graphs T_i can be computed as

$$\begin{cases} d(r_q, r_i) + 4|L_q| - (2|L_i| + \psi_D + \psi_L), & |Li| \leq |L_q| \\ d(r_q, r_i) - 2|L_q| - (-4|L_i| + \psi_D + \psi_L), & |Li| > |L_q| \end{cases}$$

where $\psi_D = |\Psi_{D_q} \cap \Psi_{D_i}|$ and $\psi_L = |\Psi_{L_q} \cap \Psi_{L_i}|$.

Algorithm 1 The Top-k Unit Tree Searching Algorithm

top-$k \leftarrow \varnothing$
for each round **do**
  for $j = 1; j \leq N + 1; j++$ **do**
    Fetch the entry T_i in the j^{th} sorted list
    if a new entry is found **then**
      Compute $\lambda(T_q, T_i)$
      if $\lambda(T_q, T_i) < \lambda_T^k$ **then**
        Update the top-k heap, i.e., $\lambda_T^k = \lambda(T_q, T_i)$
      end if
    end if
  end for
  Compute $\psi = \sum_{j=1}^{N} \min\{C_j, \Phi_j\}$
  Evaluate $\theta = 4|L_q| - (2\Phi_{N+1} + \psi)$ or $\theta = -2|L_q| - (-4\Phi_{N+1} + \psi)$
  if $\lambda_T^k \leq \theta$ **then**
    break
  end if
end for

Here, we define two aggregation functions as $\theta_1 = 4|L_q| - (2|L_i| + \psi_D + \psi_L)$ and $\theta_2 = -2|L_q| - (-4|L_i| + \psi_D + \psi_L)$. Given a query unit tree T_q, the value of $|L_q|$ is fixed. Therefore, the UTD between T_q and each T_i increases when the value of $2|L_i| + \psi_D + \psi_L$ or $-4|L_i| + \psi_D + \psi_L$ decreases. It is easy to construct sorted lists from the lower-level index following the monotonic property of these two aggregation functions. For each unit label of T_q, we fetch out the inverted list below this label and split it into two sperate lists with leaf size no larger than $|L_q|$ and larger than $|L_q|$. For processing $|L_i|$, we maintain an extra inverted list for all the profile graphs in order of increasing leaf sizes. When constructing sorted lists, the size list is also split into two groups. The group with leaf sizes no larger than $|L_q|$ should be reversely accessed in a decreasing order. As shown in Algorithm 1, given a query unit tree T_q with N distinct labels having frequencies of $(C_1, C_2, \ldots, C_N)$, we have N sorted lists and one size list. Then, we do sorted access in a round-robin schedule to each sorted list. If a new unit tree T_i is found, compute $\lambda(T_q, T_i)$. We maintain a queue of top-k unit trees with the lowest λ values. At the end of each round, we calculate the common number of unit labels as $\psi = \sum_{j=1}^{N} \min\{C_j, \Phi_j\}$, where Φ_j is the frequency value at current visited position in the j^{th} sorted list. Then, the aggregation sore is $\theta = 4|L_q| - (2\Phi_{N+1}+\psi)$ or $\theta = -2|L_q|-(-4\Phi_{N+1}+\psi)$, where Φ_{N+1} is the size value of current visited position in the size list. If the top-k values are at most equal to θ, then the halting condition is satisfied; otherwise, go to next round.

Given a query graph g_q, for each $T_q \in T(g_q)$, its top-k queue is returned from Algorithm 1 in Step 1. Then, for each unit tree T_i in the queue, a graph inverted list indexed by T_i can be directly fetched from the upper-level index. Therefore, k graph lists will be returned for each query unit tree. Obviously, graphs in sorted lists are naturally ordered in terms of UTDs according to the top-k values. With the sorted lists, the CA stage in Step 2 accesses unit trees for profile graphs using a round-robin scan. Using the summation of UTDs as an aggregation function, the halting condition and several aggregation bounds can be directly derived.

Algorithm 2 The Graph KNN Query Algorithm

top-$k \leftarrow \varnothing$
for each round **do**
  for $j = 1; j \leq M; j++$ **do**
    Fetch the entry g_i in the j^{th} sorted list
    if g_i is first found in the j^{th} sorted list **then**
      Update the value of $\zeta(g_q, g_i)$ by adding the UTD
      if $\zeta(g_q, g_i) > \lambda_T^k$ **then**
        Filter out the g_i
        continue
      end if
    end if
    Update the cost matrix between g_q and g_i
    if *scandepth*%h == 0 **then**
      for each found and unprocessed graph g_n **do**
        Compute $\lambda_L(g_q, g_n)$ using the cost matrix
        if $\lambda_L(g_q, g_n) > \lambda_T^k$ **then**
          Filter out the g_n
          continue
        end if
        Compute $\lambda(g_q, g_n)$ incrementally
        if $\lambda(g_q, g_n) < \lambda_T^k$ **then**
          Update the top-k heap, i.e., $\lambda_T^k = \lambda(g_q, g_n)$
        end if
      end for
    end if
  end for
  Evaluate $\omega = \sum_{j=1}^{M} \Phi_j$
  if $\lambda_T^k \leq \omega$ **then**
    Process all found and unprocessed graphs
    Update the top-k heap if a better result is found
    break
  end if
end for

The details of KNN query algorithm can be seen in Algorithm 2. Given a query graph g_q with M distinct unit tree sorted lists, we do sorted access in a round-robin schedule to each list. If a profile graph g_i is first found in the visited sorted list, compute $\zeta(g_q, g_i) \leftarrow \zeta(g_q, g_i) + UTD_j$ where UTD_j is the first found unit tree distance of g_i in the j^{th} sorted list. Obviously, $\zeta(g_q, g_i)$ is a lower bound of $\lambda(g_q, g_i)$. If $\zeta(g_q, g_i)$ is greater than the top-k values, the graph g_i can be safely filtered out. At each depth h of visited lists, for all found and unprocessed graphs, we compute the partial mapping distance λ_L. We maintain a queue of top-k graphs with the lowest λ values. At the end of each round, we evaluate the aggregation score $\omega = \sum_{j=1}^{M} \Phi_j$ where Φ_j is the UTD value at current visited position in the j^{th} sorted list. If the top-k values are at most equal to ω, then the halting condition is satisfied; otherwise, go to the next round.

Algorithm 3 The Graph Range Query Algorithm

```
resultset ← ∅ and τ
for each round do
  for j = 1; j ≤ M; j++ do
    Fetch the entry g_i in the j^th sorted list
    if g_i is first found in the j^th sorted list then
      Update the value of ζ(g_q, g_i) by adding the UTD
      if ζ(g_q, g_i) > τ then
        Filter out the g_i
        continue
      end if
    end if
    Update the cost matrix between g_q and g_i
    if scandepth%h == 0 then
      for each found and unprocessed graph g_n do
        Compute λ_L(g_q, g_n) using the cost matrix
        if λ_L(g_q, g_n) > τ then
          Filter out the g_n
          continue
        end if
        Compute λ(g_q, g_n) incrementally
        if λ(g_q, g_n) ≤ τ then
          Add the g_n into resultset
        end if
      end for
    end if
  end for
  Evaluate ω = Σ_{j=1}^{M} Φ_j
  if τ ≤ ω then
    Process all found and unprocessed graphs
    Filter out those graphs with mapping distance larger
    than τ
    break
  end if
end for
```

The details of range query algorithm is shown in Algorithm 3. The pruning strategy is similar to Algorithm 2. The difference is that we filter out graphs with mapping distances larger than the threshold value of τ, instead of maintaining a top-k queue. Here, we omit all proofs for the correctness of three algorithms as they were shown in previous work [46].

VI. THE LAZY LEARNING APPROACH

We employ the dynamic time warping [47] to measure the similarity between two evolving graphs, denoted by the graph alignment distance.

Definition 5 [Graph Alignment Distance (GAD)]: Given two evolving graphs G^1 and G^2 with their multisets of profile graphs, and a bijection $P : G^1 \rightarrow G^2$, the alignment distance between G^1 and G^2 denoted by $\lambda(G^1, G^2)$ is computed as

$$\lambda(G^1, G^2) = \min_P \sum_{g_i^1 \in G^1} \lambda(g_i^1, P(g_i^1)).$$

Algorithm 4 The Dynamic Programming Algorithm

```
M ← array[0..n, 0..m]
for i = 1; i ≤ n; i++ do
  M[i, 0] ← ∞
end for
for i = 1; i ≤ m; i++ do
  M[0, i] ← ∞
end for
for i = 1; i ≤ n; i++ do
  for j = 1; j ≤ m; j++ do
    cost ← λ(g_i^1, g_j^2)
    min ← min(M[i−1, j], M[i, j−1], M[i−1, j−1])
    M[i, j] ← cost + min
  end for
end for
λ(G^1, G^2) ← M[n, m]
```

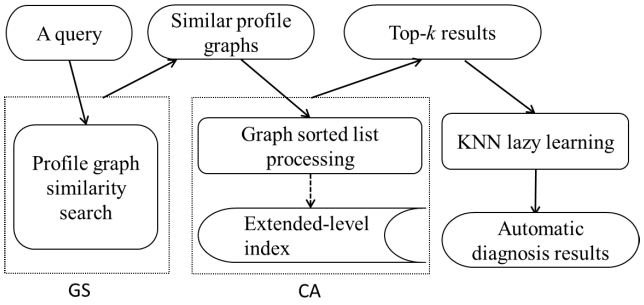

FIGURE 8. The framework of lazy learning approach.

The computation of $\lambda(G^1, G^2)$ is equivalent to solving the sequence alignment problem, which is typically solved using the dynamic time warping algorithm aiming at finding the minimum weight matching in a given cost matrix. Given two multisets of profile graphs $\{g_1^1, g_2^1, \ldots, g_n^1\}$ and $\{g_1^2, g_2^2, \ldots, g_m^2\}$, we construct an $n \times m$ weighted matrix M with each weight computed as $\lambda(g_i^1, g_j^2)$ between each pair of profile graphs. We apply the dynamic programming to finding the optimal alignment as shown in Algorithm 4.

Figure 8 shows the framework of proposed lazy learning approach. We first extend the two-level inverted index to three levels for supporting efficient prediction analysis. We further build an extended-level index to store the profile graphs for each evolving graph. Given a set of evolving graphs and their sequences of profile graphs, the index is made up of two main parts: an index for all distinct profile graphs, and an inverted list below each profile graph. Each entry in the inverted lists contains the evolving graph identity and the frequency of the corresponding profile graph.

Given a query G^q, for each profile graph $g_i^q \in G^q$, its top-k queue are returned from Algorithm 2. Then, all inverted list indexed by the top-k results can be directly fetched from the extended-level index, and combined into one sorted list. Obviously, evolving graphs in the sorted list are naturally ordered in terms of GMDs according to the top-k values.

Algorithm 5 The KNN Learning Algorithm

top-$k \leftarrow \varnothing$
for each round **do**
  for $j = 1; j \leq n; j++$ **do**
    Fetch the entry G^i in the j^{th} sorted list
    if G^i is first found in the j^{th} sorted list **then**
      Update the value of $\zeta(G^q, G^i)$ with GMD
      if $\zeta(G^q, G^i) > \lambda_T^k$ **then**
        Filter out the G^i
        continue
      end if
    end if
    Update the cost matrix between G^q and G^i
    if $scandepth\%h == 0$ **then**
      for each found and unprocessed graph G^x **do**
        Compute $\lambda_L(G^q, G^x)$ using the cost matrix
        if $\lambda_L(G^q, G^x) > \lambda_T^k$ **then**
          Filter out the G^x
          continue
        end if
        Compute $\lambda(G^q, G^x)$ incrementally
        Update the top-k queue with a better result
      end for
    end if
  end for
  Evaluate $\omega = \sum_{j=1}^{n} \Phi_j$
  if $\lambda_T^k \leq \omega$ **then**
    Process all found and unprocessed graphs
    Update the top-k queue if better results are found
    break
  end if
end for
Predict G^q using a majority vote of top-k results

With the sorted lists, we use the CA algorithm to access profile graphs using a round-robin scan. Using the summation of GMDs as an aggregation function, the halting condition and several aggregation bounds can be directly derived.

The details of KNN learning algorithm can be seen in Algorithm 5. Given a query graph G^q with n sorted lists, we do sorted access in a round-robin schedule to each list. If an entry G^i is first found in the visited sorted list, compute $\zeta(G^q, G^i) \leftarrow \zeta(G^q, G^i) + GMD_j$ where GMD_j is the first found graph mapping distance of G^i in the j^{th} sorted list. Obviously, $\zeta(G^q, G^i)$ is a lower bound of $\lambda(G^q, G^i)$. If $\zeta(G^q, G^i)$ is greater than the top-k values, the graph G^i can be safely filtered out. At each depth h of visited lists, for all found and unprocessed graphs, we compute the partial alignment distance λ_L and extract distance λ for further pruning. We maintain a top-k queue with the lowest λ values. At the end of each round, we evaluate the aggregation score $\omega = \sum_{j=1}^{n} \Phi_j$ where Φ_j is the GMD value at current visited position in the j^{th} sorted list. If the top-k values are at most equal to ω, then the halting condition is satisfied; otherwise, go to the next round. By obtaining the top-k results, G^q is predicted by a majority vote of its k neighbors [48].

VII. EXPERIMENTAL STUDY

This section presents the results of the extensive performance study of our approach. All the codes are implemented in C++. We conduct the experiments on a server with 32GB memory, running Centos 5.6.

A. DATASETS

Two real datasets are used to evaluate the proposed approach.

MIMIC-III is a public EMR database [49] consisting of 46,776 records from patients who are admitted to the hospital between 2001 and 2012. This dataset includes patients' lab tests, medications, procedures, demographics, and vital sign measurements, etc. In this dataset, we use each admission as a record, which refers to a patient's one visit to the hospital. For each admission, we reserve one disease diagnosis code.

OH is a real-world longitudinal EMR dataset from an Orthopaedic Hospital in China. We conduct experiments in a sample dataset of the OH dataset with 12,000 admissions in OH from 2010 to 2014. Patients' medical data such as demographics, diagnoses, lab tests, medications and procedures are collected in this dataset. We processed the data and stored them using similar schemas used by MIMIC-III. For each admission, there is also one disease diagnosis code.

In this paper, we perform two tasks: (1) Medical case search by a query sample is to return similar cases similar disease cases to the query patient; (2) Diagnosis by category is to predict the disease category of a query patient.

B. MEDICAL CASE SEARCH EVALUATION

For both datasets, we randomly selected 1000 profile graphs as queries to do the medical case search. The running time is the average query time of all queries, and the accuracy is the average precision of all queries.

1) SETTINGS

We denote our proposed profile graph search methods respectively as **PGS-KNN** for KNN query and **PGS-range** for range query. For range query, our work is the first focus on this problem. We evaluate it by setting the range threshold value of τ as 1, 5, 10, 15, 20.

We compared our **PGS-KNN** with a baseline algorithm, namely **Feature-KNN**. The baseline represents a patient's medical data as a feature vector and computes the cosine similarity between the queries and all samples. We set the parameter k value as 10, 50, 100, 150, 200.

2) EVALUATION WITH RUNNING TIME

Figure 9 (y-axis is log-scaled) shows that both our proposed medical search algorithms are fast enough to support real applications. The average query time for medical case searching in our experiments is less than 2.5 seconds even for returning a top-200 results. Although the **PGS-KNN** is slower compared with the **Feature-KNN**, it makes sense to

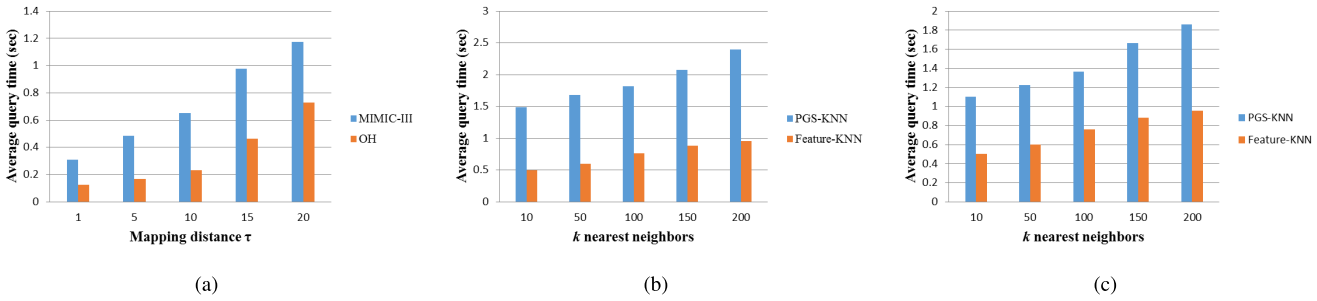

FIGURE 9. Evaluation results on running time. (a) PGS-range. (b) MIMIC-III. (c) OH.

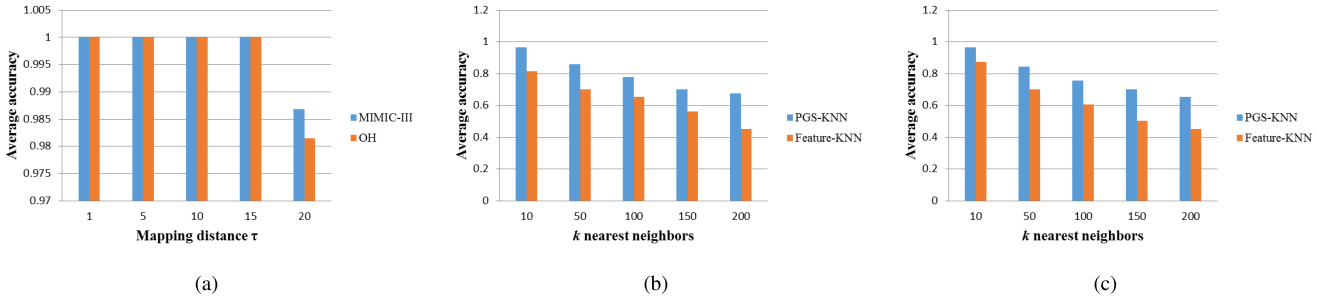

FIGURE 10. Evaluation results on accuracy. (a) PGS-range. (b) MIMIC-III. (c) OH.

sacrifice a little more time to obtain a higher accuracy in real applications, as **PGS-KNN** does. The reason is that the accuracy of the KNN results will directly affect the accuracy of final diagnosis. We later show the accuracy comparison evaluation in Figure 10.

3) EVALUATION WITH ACCURACY

Figure 10 shows the accuracy performance of our methods compared to the baseline. Figure 10 (a) illustrates the very high accuracy achieved by the **PGS-range**, even when the range threshold value τ grows to a large number of 20.

For both two real datasets, our proposed **PGS-KNN** achieves higher accuracy value compared against the **Feature-KNN**, indicating that our consideration of correlation between medical features helps resolve the bias in EMR data and contributes to more accurate medical case search. The results can be seen in Figure 10 (b) and (c).

C. BENEFITS FOR AUTOMATIC DIAGNOSIS

We compare our proposed method with the baseline method in predicting disease category in two real datasets illustrated in Figure 11. For both datasets, our method outperforms the baseline. This observation indicates that analytical tasks can benefit from considering the feature correlations which reduce the information loss caused by feature vector based methods. **PGS-KNN** can achieve higher AUROC value of this task, indicating that our proposed dynamic graph models contributes to more accurate automatic diagnosis.

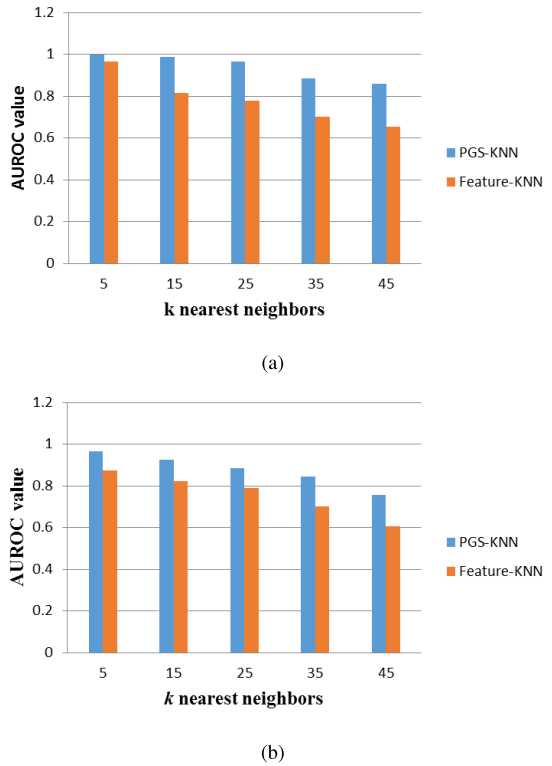

FIGURE 11. Diagnosis by category results. (a) MIMIC-III. (b) OH.

D. EMPIRICAL STUDY

An empirical study was performed with the help of our previous implemented crowdsourcing based expert Q&A system [10]. In this experiment, we utilized top-10 results

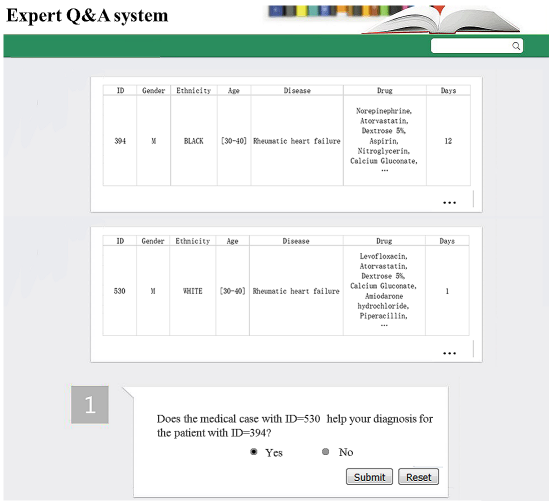

FIGURE 12. Expert Q&A platform.

returned from our algorithm. As shown in Figure 12, given each result and the query medical case, we designed an appropriate simple yes or no question, which was distributed into the expert Q&A system for collecting experts' answers. Note that, the experts in our system are doctors from real hospitals.

Three types of questions are generated for the expert Q&A platform including:

- Q1: Does a returned medical case help the doctors' diagnoses on the query medical case?
- Q2: Does a returned medical case have the same disease as the query medical case?
- Q3: Does the patient of the query medical case have the same disease as the one of the returned medical case?

Experts can choose "yes" or "no" to answer the questions. We statistically analyzed the collected answers with the average accuracy rate. The average accuracy rate is defined in Equation (1).

$$\begin{aligned} P_{ave} &= Average\,(P_1, P_2, \ldots, P_i, \ldots, P_n) \\ P_i &= \frac{|X_i|}{|Y_i|} \end{aligned} \tag{1}$$

X_i is a number of collected "yes" answers from the expert system for a question i, and Y_i is the total number of collected experts' responses. Finally the average accuracy we obtained is about 62%.

VIII. CONCLUSION

In this paper, we first propose a novel modeling technique to represent complex medical data as sequence of evolving graphs. The proposed model could achieve better expressivity of medical data. Then, we develop a novel lazy learning algorithm to support efficient automatic diagnosis based on evolving graph similarity search. We also construct a multi-level inverted index to speed up the graph similarity search, and support efficient prediction analysis for automatic diagnosis. The experimental results show that the proposed approach has better performance compared against the baseline. In the future, we will consider to employ the GP model in graph sequence prediction. Different from typical algorithms attempting to eagerly train a global GP model on the entire dataset, we may focus on developing a semi-lazy learning approach, which is a hybrid of the GP learning and graph similarity search.

REFERENCES

[1] J. Sun and C. K. Reddy, "Big data analytics for healthcare," in *Proc. ACM SIGKDD*, 2013, p. 1525.
[2] J. Q. Chen, R. D. Zhang, and A. Chen, "Challenges of big data analytics applications in healthcare," *Commun. ICISA*, vol. 16, no. 1, pp. 21–35, 2015.
[3] H. Asri, H. Mousannif, H. A. Moatassime, and T. Noel, "Big data in healthcare: Challenges and opportunities," in *Proc. Int. Conf. Cloud Technol. Appl.*, 2015, pp. 1–7.
[4] C. Lee *et al.*, "Big healthcare data analytics: Challenges and applications," in *Scalable Computing and Communications*. 2017, pp. 11–41.
[5] C. Friedman, P. O. Alderson, J. H. Austin, J. J. Cimino, and S. B. Johnson, "A general natural-language text processor for clinical radiology," *J. Amer. Med. Inf. Assoc.*, vol. 1, no. 2, pp. 161–174, 1994.
[6] N. H. Phuong, "Fuzzy set theory and medical expert systems: Survey and model," in *Theory and Practice of Informatics*. 1995, pp. 431–436.
[7] G. K. Savova *et al.*, "Mayo clinical text analysis and knowledge extraction system (cTAKES): Architecture, component evaluation and applications," *J. Amer. Med. Informat. Assoc. Jamia*, vol. 17, no. 5, pp. 507–513, 2010.
[8] H. Alemdar and C. Ersoy, "Wireless sensor networks for healthcare: A survey," *Comput. Netw.*, vol. 54, no. 15, pp. 2688–2710, Oct. 2010.
[9] Z. J. Ling *et al.*, "GEMINI: An integrative healthcare analytics system," *VLDB Endowment*, vol. 7, no. 13, pp. 1766–1771, 2014.
[10] Z. Xu, X. Wang, Y. Chen, Y. Pan, M. Wu, and M. Xiong, "ADDS: An automated disease diagnosis-aided system," *Web Technologies and Applications*. 2016, pp. 556–560.
[11] D. T. Gunter and P. N. Terry, "The emergence of national electronic health record architectures in the United States and Australia: Models, costs, and questions," *J. Med. Internet Res.*, vol. 7, no. 1, p. e3, 2005.
[12] Z. Kuang, J. Thomson, M. Caldwell, P. Peissig, R. Stewart, and D. Page, "Computational drug repositioning using continuous self-controlled case series," in *Proc. ACM SIGKDD*, 2016, pp. 491–500.
[13] G. Stalidis, A. Prentza, I. N. Vlachos, S. Maglavera, and D. Koutsouris, "Medical support system for continuation of care based on xml web technology," *Int. J. Med. Informat.*, vol. 64, nos. 2–3, p. 385, 2001.
[14] T. Stiehl, M. Führer, T. Roser, C. Kunzmann, and A. Schmidt, "Describing spiritual care within pediatric palliative care. An ontology-based method for qualitative research," in *Proc. Portugal Congr. Eur. Assoc. Palliative Care*, 2011, pp. 164–172.
[15] M. Singh and K. Kaur, "SQL2NEO: Moving health-care data from relational to graph databases," in *Proc. Adv. Comput. Conf.*, 2015, pp. 721–725.
[16] M. S. A. Malik, S. Sulaiman, and M. F. A. Malik, "Relational factors of EHR database in visual analytic systems for public health care units," in *Proc. Int. Multi Topic Conf.*, 2014, pp. 161–172.
[17] M. Hassan, A. Coulet, and Y. Toussaint, "Learning subgraph patterns from text for extracting disease-symptom relationships," in *Proc. Int. Conf. Interact. Between Data Mining Natural Lang. Process.*, 2014, pp. 81–96.
[18] F. Wang, N. Lee, J. Hu, J. Sun, and S. Ebadollahi, "Towards heterogeneous temporal clinical event pattern discovery: A convolutional approach," in *Proc. ACM SIGKDD*, 2012, pp. 453–461.
[19] X. Wang, D. Sontag, and F. Wang, "Unsupervised learning of disease progression models," in *Proc. ACM SIGKDD*, 2014, pp. 85–94.
[20] Z. Che, D. Kale, W. Li, M. T. Bahadori, and Y. Liu, "Deep computational phenotyping," in *Proc. ACM SIGKDD*, 2015, pp. 507–516.
[21] K. Zheng, J. Gao, K. Y. Ngiam, B. C. Ooi, and W. L. J. Yip, "Resolving the bias in electronic medical records," in *Proc. ACM SIGKDD*, 2017, pp. 2171–2180.
[22] T. Tran, D. N. Tu, D. Phung, and S. Venkatesh, "Learning vector representation of medical objects via EMR-driven nonnegative restricted Boltzmann machines (eNRBM)," *J. Biomed. Informat.*, vol. 54, pp. 96–105, Apr. 2015.
[23] I. M. Baytas, K. Lin, F. Wang, A. K. Jain, and J. Zhou, "Stochastic convex sparse principal component analysis," *EURASIP J. Bioinformat. Syst. Biol.*, vol. 2016, no. 1, pp. 1–11, 2016.

[24] T. A. Lasko, J. C. Denny, and M. A. Levy, "Computational phenotype discovery using unsupervised feature learning over noisy, sparse, and irregular clinical data," *PLoS ONE*, vol. 8, no. 6, p. e66341, 2013.
[25] L. A. Celi, A. J. Zimolzak, and D. J. Stone, "Dynamic clinical data mining: Search engine-based decision support," *JMIR Med. Informat.*, vol. 2, no. 1, p. e13, 2014.
[26] A. Lotem, "Optimal aggregation algorithms for middleware," *J. Comput. Syst. Sci.*, vol. 66, no. 4, pp. 614–656, 2003.
[27] C. E. Kuziemsky *et al.*, "Ontology-based information integration in health care: A focus on palliative care," in *Proc. 11th Int. Workshop Softw. Technol. Eng. Pract.*, 2004, pp. 164–172.
[28] F. Wu, H. Zhang, and Y. Zhuang, "Learning semantic correlations for cross-media retrieval," in *Proc. IEEE Int. Conf. Image Process.*, Oct. 2007, pp. 1465–1468.
[29] Y. Yang, Y.-T. Zhuang, F. Wu, and Y.-H. Pan, "Harmonizing hierarchical manifolds for multimedia document semantics understanding and cross-media retrieval," *IEEE Trans. Multimedia*, vol. 10, no. 3, pp. 437–446, Apr. 2008.
[30] Y. T. Zhuang, Y. Yang, and F. Wu, "Mining semantic correlation of heterogeneous multimedia data for cross-media retrieval," *IEEE Trans. Multimedia*, vol. 10, no. 2, pp. 221–229, Feb. 2008.
[31] Y. Yang, D. Xu, F. Nie, J. Luo, and Y. Zhuang, "Ranking with local regression and global alignment for cross media retrieval," in *Proc. Int. Conf. Multimedia*, 2009, pp. 175–184.
[32] X. Mao, B. Lin, D. Cai, X. He, and J. Pei, "Parallel field alignment for cross media retrieval," in *Proc. ACM Int. Conf. Multimedia*, 2013, pp. 897–906.
[33] X. Wang, "Developing 3-in-1 index structures on complex structure similarity search," Ph.D. dissertation, Nat. Univ. Singapore, Singapore, 2013.
[34] J. Zhou *et al.*, "Generic inverted index on the GPU," Tech. Rep., 2016.
[35] S. P. Ahuja, S. Mani, and J. Zambrano, "A survey of the state of cloud computing in healthcare," *Netw. Commun. Technol.*, vol. 1, no. 2, pp. 12–19, 2012.
[36] T. Ermakova, J. Huenges, K. Erek, and R. Zarnekow, "Cloud computing in healthcare—A literature review on current state of research," in *Proc. Amer. Conf. Inf. Syst.*, 2013, pp. 1–9.
[37] A. K. Gupta and K. S. Mann, "Sharing of medical information on cloud platform-a review," *J. Comput. Eng.*, vol. 16, no. 2, pp. 8–11, 2014.
[38] L. Griebel *et al.*, "A scoping review of cloud computing in healthcare," *BMC Med. Informat. Decis. Making*, vol. 15, no. 1, p. 17, 2015.
[39] J. Hu, A. Perer, and F. Wang, "Data driven analytics for personalized healthcare," in *Healthcare Information Management Systems*. 2016, pp. 529–554.
[40] K. Lin, M. Wu, X. Wang, and Y. Pan, "MEDLedge: A Q&A based system for constructing medical knowledge base," in *Proc. Int. Conf. Comput. Sci. Educ.*, 2016, pp. 485–489.
[41] Y. Guo, Z. Pan, and J. Heflin, "LUBM: A benchmark for owl knowledge base systems," *Web Semantics, Sci., Services Agents World Wide Web*, vol. 3, no. 2, pp. 158–182, 2005.
[42] J. Mora and O. Corcho, "Towards a systematic benchmarking of ontology-based query rewriting systems," in *The Semantic Web*, vol. 8219. 2013, pp. 376–391.
[43] H. Dinari, "A survey on graph queries processing: Techniques and methods," *Int. J. Comput. Netw. Inf. Secur.*, vol. 9, no. 4, pp. 48–56, 2017.
[44] Z. Zeng, A. K. H. Tung, J. Wang, J. Feng, and L. Zhou, "Comparing stars: On approximating graph edit distance," *VLDB Endowment*, vol. 2, no. 1, pp. 25–36, 2009.
[45] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Res. Logistics*, vol. 52, no. 1, pp. 7–21, 2010.
[46] X. Wang, X. Ding, A. K. H. Tung, S. Ying, and H. Jin, "An efficient graph indexing method," in *Proc. IEEE Int. Conf. Data Eng.*, Apr. 2012, pp. 210–221.
[47] S. Salvador and P. Chan, "Toward accurate dynamic time warping in linear time and space," in *Proc. KDD Workshop Mining Temporal Sequential Data*, 2004, pp. 70–80.
[48] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *Amer. Statist.*, vol. 46, no. 3, pp. 175–185, 1992.
[49] A. E. W. Johnson *et al.*, "MIMIC-III, a freely accessible critical care database," *Scientific Data*, vol. 3, p. 160035, May 2016.

XIAOLI WANG received the B.S. degree in computer science from the Northeastern University of China and the Ph.D. degree in computer science from the National University of Singapore. She is currently an Assistant Professor at the School of Software, Xiamen University, Xiamen, China. Her research interests mainly focus on database and data mining relevant issues, including indexing and query processing on complex structures, data cleaning and integration, textual data processing, big healthcare data, and social media computing.

YUAN WANG received the Ph.D. degree in optimization and simulation with specialization in transportation and manufacturing system from the National University of Singapore in 2012. She is currently a Senior Research Fellow at the Department of Industrial and Systems Engineering, National University of Singapore. Her major research interests include mathematical modeling, machine learning, complex system simulation, and optimization heuristics. She is currently a key leader involved in smart city projects in the Center of Next Generation Logistics co-founded by the National University of Singapore and the Georgia Institute of Technology.

CHUCHU GAO received the B.S. degree in software engineering from the School of Software, Xiamen University, Xiamen, China, where she currently pursuing the master's degree. Her major research interests include electronic commerce and big data mining relevant issues, including textual data processing and big healthcare data.

KUNHUI LIN received the B.S. and M.S. degrees from the School of Information Science and Engineering, Xiamen University, Xiamen, China. He is currently a Professor at the School of Software, Xiamen University. He is currently the Director of the Digital Fujian Urban Public Safety Big Data Research Institute. His research interests include urban public safety, big data and cloud computing, intelligent information processing, e-commerce, and network multimedia relevant issues.

YADI LI received the M.D. degree in dermatology from Capital Medical University in 2012. She is currently an Attending Doctor with the Department of Dermatology, Beijing Tongren Hospital. Her sub-specialties are skin cosmetic medicine and dermatology surgery.