A Methodology to Generate Virtual Patient Repositories
Uri Kartoun PhD, Massachusetts General Hospital / Harvard Medical School

ABSTRACT
Electronic medical records (EMR) contain sensitive personal information. For example, they may include details about infectious diseases, such as human immunodeficiency virus (HIV), or they may contain information about a mental illness. They may also contain other sensitive information such as medical details related to fertility treatments. Because EMRs are subject to confidentiality requirements, accessing and analyzing EMR databases is a privilege given to only a small number of individuals. Individuals who work at institutions that do not have access to EMR systems have no opportunity to gain hands-on experience with this valuable resource. Simulated medical databases are currently available; however, they are difficult to configure and are limited in their resemblance to real clinical databases. Generating highly accessible repositories of virtual patient EMRs while relying only minimally on real patient data is expected to serve as a valuable resource to a broader audience of medical personnel, including those who reside in underdeveloped countries.

Keywords: simulation in healthcare, electronic medical records, electronic health records

INTRODUCTION
The importance of patient privacy has been thoroughly emphasized by governmental resources such as the HIPAA Privacy Rule and by academia [1–3]. Numerous strategies to maintain patients’ privacy have been developed [4–8]; however, the medical profession is not yet able to guarantee full protection of privacy while providing detailed information about each patient [9].

Cohorts assembled from electronic medical records (EMRs) represent a powerful resource to study disease complications at a population level. Recent studies have demonstrated the usefulness of EMR analysis for discovering or confirming outcome correlations, subcategories of disease, and adverse drug events [10–16]. Due to confidentiality restrictions, accessing and analyzing EMR databases is a privilege given to only a small number of individuals. Individuals who work in institutions that do not have access to EMR systems cannot experiment with such valuable resources. When professors wish to teach a biomedical informatics course focused on EMR technology, they cannot distribute real EMR data among their students.

Simulated medical databases [e.g., 17, 18] and open EMR platforms [19] are currently available; however, no confidentiality-free massive scale longitudinal EMR databases have yet been algorithmically created. Virtual
patient repositories that bear a high degree of resemblance to real patient databases while relying only minimally on real patient data are expected to serve as a valuable resource for medical professionals in training, and to accelerate health care research and development.

The aim of this study is to develop a novel methodology for creating virtual patient repositories. I demonstrate that a method entailing minimal configuration requirements can generate nonconfidential artificial EMR databases that could be used to practice statistical and machine-learning algorithms. I further demonstrate the potential broad public interest in the availability of this technology.

MATERIALS AND METHODS
The process of generating a virtual patient repository was based on preconfiguration of population-level and patient-level characteristics. First, an object-oriented program acquired a population-level configuration to generate patient objects. Next, the program created a clinical profile for each patient, including admissions associated with chief complaints and laboratory measurements. Finally, the patient objects were stored in a database. Following this methodology, three databases of 100, 10,000, and 100,000 virtual patients were created.

Population-level configuration
Population-level configuration specifies the number of individual records that will be generated and defines preconfigured values for demographic characteristics. Demographic characteristics include gender, marital status, major language, ethnicity, date of birth, and income level. Configuring categorical variables defines several potential values for the variable and the percentage of the population with the value. For example, in a population of \( n = 100,000 \) individuals, a potential configuration for ethnicity would be 49% white, 23% Asian, 15% African American, and the remainder unknown. For continuous variables such as age, the population percentages for several ranges of date of birth are defined. For example, dates of birth in the range of 1940 to 1950 were randomly created for 15% of the population. The configuration tables are presented in Supplemental Table 1(a–b).

Generating a virtual patient repository
Having acquired the population-level configuration, the program generated \( n \) objects each representing one virtual patient. Each such object is associated with distinct demographic characteristics (e.g., a certain patient might be an English-speaking, Asian, single woman who was born on May 17, 1982). For each such object, the
program randomly assigns an integer representing the number of admissions associated with that patient throughout his or her lifetime. Each admission is associated with additional details, including randomly generated length of stay (in days) and start and end dates. Further, each admission is associated with laboratory measurements and a chief complaint randomly selected from a list of International Classification of Diseases, Tenth Revision, Clinical Modification (ICD-10-CM) codes. Laboratory values are based on 35 common types—for instance, sodium levels, creatinine levels, or platelet count. The program generates for each admission multiple dates and times for each laboratory type; values are bounded by predefined ranges for each laboratory type. For example, a patient may be associated with a 3-day admission during which creatinine was measured 6 times throughout the admission. The configuration table for laboratory types and ranges is presented in Supplemental Table 2. An example for the EMR lifespan of one virtual patient is presented in Figure 1.

![Figure 1. An example of the longitudinal EMR of a virtual patient.](image)

**RESULTS**

**A virtual patient repository**

Table 1 summarizes several characteristics of the largest cohort, which comprised 100,000 virtual patients. Each virtual patient in the cohort is associated with 1 to 10 admissions; each admission lasts from 1 to 20 days and is associated with a single chief complaint. All values are associated with a date and time. Each admission record also includes multiple measurements of common laboratory tests (see Supplemental Table 2). The number of admissions per patient, the length of stay per admission, and the laboratory values were randomly generated; however, they are sampled from predefined ranges of values. For example, a patient’s age could not exceed 95 years old as of January 1, 2015, and a glucose measurement could only be in the range of 60–140 mg/dL. Only chief complaints common for both men and women were allowed in a cohort. In total, the database contained
1.4 GB of data, representing 100,000 virtual patients associated with 361,760 admissions and 107,535,387 total laboratory measurements.

Table 1. A virtual patient repository of 100,000 patients.

<table>
<thead>
<tr>
<th>Variable and category</th>
<th>Patients (n = 100,000)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean age as of 1/1/2015, years (SD)</td>
<td>57.8 (17.3)</td>
</tr>
<tr>
<td>Gender (%)</td>
<td></td>
</tr>
<tr>
<td>Female</td>
<td>52.0</td>
</tr>
<tr>
<td>Ethnicity (%)</td>
<td></td>
</tr>
<tr>
<td>White</td>
<td>49.0</td>
</tr>
<tr>
<td>Asian</td>
<td>23.0</td>
</tr>
<tr>
<td>African American</td>
<td>15.0</td>
</tr>
<tr>
<td>Unknown</td>
<td>13.0</td>
</tr>
<tr>
<td>Mean number of admissions per patient, days (SD)</td>
<td>3.6 (1.5)</td>
</tr>
<tr>
<td>Mean length of stay (SD)</td>
<td>11.0 (5.2)</td>
</tr>
<tr>
<td>% Population with length of follow-up (years)</td>
<td></td>
</tr>
<tr>
<td>0 - 9</td>
<td>13.1</td>
</tr>
<tr>
<td>10 - 15</td>
<td>9.3</td>
</tr>
<tr>
<td>&gt; 15</td>
<td>77.6</td>
</tr>
<tr>
<td>Population below poverty (%)</td>
<td>21.6</td>
</tr>
<tr>
<td>Comorbidities; Prevalence (%)</td>
<td></td>
</tr>
<tr>
<td>Malignant neoplasm</td>
<td>41.4</td>
</tr>
<tr>
<td>Rheumatoid arthritis</td>
<td>25.6</td>
</tr>
<tr>
<td>Diabetes (type I or II)</td>
<td>24.4</td>
</tr>
<tr>
<td>Renal complications</td>
<td>17.0</td>
</tr>
<tr>
<td>Coronary artery disease</td>
<td>7.0</td>
</tr>
<tr>
<td>Laboratory values (Mean; SD)</td>
<td></td>
</tr>
<tr>
<td>Blood urea nitrogen (mg/dL)</td>
<td>17.5; 7.2</td>
</tr>
<tr>
<td>Platelets (k/cumm)</td>
<td>284.9; 95.3</td>
</tr>
<tr>
<td>Creatinine (mg/dL)</td>
<td>0.9; 0.2</td>
</tr>
<tr>
<td>Albumin (gm/dL)</td>
<td>4.2; 1.0</td>
</tr>
<tr>
<td>Lymphocytes (%)</td>
<td>25; 5.8</td>
</tr>
</tbody>
</table>

**DISCUSSION**

In April 2015, the three generated cohorts were made publically available for downloading at the dedicated website www.emrbots.org. Since its launch, the site has been visited by more than 5,000 individuals; approximately 1,000 visitors registered with their full names, institution names, and e-mail addresses. The represented institutions included top American universities, pharmaceutical companies, and governmental agencies such as the Centers for Disease Control and Prevention (CDC).

The virtual patient repositories can be instantaneously downloaded to allow users to practice algorithms on EMR-like data with a primary objective of improving their technical skills. These skills may be used in the future when...
the individuals are granted access to real EMR. Using the repositories would not require installation of any software. The repositories contain raw data only and are provided as textual flat files; thus they are independent of any specific operating system, and may be used with a wide variety of database systems. Further, the repositories can be used even on low-performance computers, as they can be run on open-source software products—such as R and MySQL—which have minimal requirements to install. The repositories could also be deployed on existing open-source EMR platforms [19]. With no confidentiality restrictions, the cohorts can be distributed to a class of any size. For example, professors could distribute the records to a class of 100 students, all of whom could then use them to practice algorithms and to build models; at the end of the course, these students would have greater experience and capabilities, and would thus be more attractive to employers in the field of medical informatics.

While the cohorts are useful for practicing machine-learning algorithms, they cannot serve as effective resources to assess real patient outcome scenarios (e.g., 30-day readmission prediction or disease prognosis) because their creation process does not take into account the complex time-dependent interactions between the factors associated with real patients. Developing algorithms to create virtual patient repositories that will reliably mimic real EMR presents is a tremendous challenge because of it necessitates approaches that will populate databases with a combination of linear and nonlinear associations between all medical elements, as well as with random associations. The algorithms required would comprise both individual-level and population-level assumptions and apply an intelligent functionality to assign acceptable temporal differences among all medical events. A virtual patient repository of nonalcoholic fatty liver disease patients, for example, would need to include assumptions about inverse correlations of albumin levels and sodium levels with cardiovascular disease [20], while virtual congestive heart failure (CHF) patients would need to be associated with a high prevalence of diuretic use, advanced age, and a high prevalence of associated comorbidities, such as renal failure [21].

A fundamental requirement is to define appropriate performance measures to assess the quality of the generated virtual data. Achieving a high level of resemblance between virtual and real patient data will necessitate development of methods to identify similarities among patients. Such methods are available; however, they focus either on several predefined variables [e.g., 22] based on analyzing images [e.g., 23] or on genetic similarities in patients [e.g., 24]. There are as yet no available methods that consider all the complex relationships among EMR variables. Current strategies for identifying groups of patients with similar time-dependent characteristics
include time-series analysis methods [e.g., 25–27] and clustering techniques [e.g., 11, 28–32]. Such methods are based on either the supervised learning paradigm (which requires extracting manually designed features) or on the unsupervised learning paradigm (in which candidate variables are computationally proposed). The task of identifying patients with similar characteristics is multidimensional. For example, two patients might have similar diagnoses (e.g., both had a recent heart failure episode, or both are diabetic); however, they might vary in many characteristics (e.g., comorbidities or laboratory measurements). Identifying distinct groups of patients with similar laboratory trends, correlated with comorbidities and medications changing over time and with demographics, is expected to accelerate the development of algorithms that will more realistically create virtual patients.

In conclusion, this study describes a novel methodology for generating confidentiality-free virtual patient repositories. The methodology may serve as a foundation to further generate large, longitudinal artificial EMR databases that highly resemble real patient records. Unlike other methods that typically obscure or shift real patients’ data elements, the proposed methodology is invulnerable in terms of security because it does not rely on real data elements pulled from an existing EMR; therefore, it is not associated with privacy concerns in regard to individuals’ sensitive data.

REFERENCES


Supplemental Table 1. Population-level configuration

(a) Categorical variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>Weight (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gender</td>
<td>Male</td>
<td>48</td>
</tr>
<tr>
<td></td>
<td>Female</td>
<td>52</td>
</tr>
<tr>
<td>Marital status</td>
<td>Single</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td>Married</td>
<td>45</td>
</tr>
<tr>
<td></td>
<td>Divorced</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>Separated</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Widowed</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Unknown</td>
<td>6</td>
</tr>
<tr>
<td>Language</td>
<td>English</td>
<td>64</td>
</tr>
<tr>
<td></td>
<td>Icelandic</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>Spanish</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>Unknown</td>
<td>6</td>
</tr>
<tr>
<td>Ethnicity</td>
<td>African American</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>Asian</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>White</td>
<td>49</td>
</tr>
<tr>
<td></td>
<td>Unknown</td>
<td>13</td>
</tr>
</tbody>
</table>

(b) Continuous variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Weight (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date of birth</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/1/1920</td>
<td>1/1/1930</td>
<td>8</td>
</tr>
<tr>
<td>1/1/1930</td>
<td>1/1/1940</td>
<td>9</td>
</tr>
<tr>
<td>1/1/1940</td>
<td>1/1/1950</td>
<td>15</td>
</tr>
<tr>
<td>1/1/1950</td>
<td>1/1/1960</td>
<td>19</td>
</tr>
<tr>
<td>1/1/1960</td>
<td>1/1/1970</td>
<td>24</td>
</tr>
<tr>
<td>1/1/1970</td>
<td>1/1/1980</td>
<td>15</td>
</tr>
<tr>
<td>1/1/1980</td>
<td>1/1/1990</td>
<td>10</td>
</tr>
<tr>
<td>Population percentage</td>
<td></td>
<td></td>
</tr>
<tr>
<td>below poverty</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>10.1</td>
<td>80</td>
</tr>
<tr>
<td></td>
<td>80.1</td>
<td>10</td>
</tr>
</tbody>
</table>

Supplemental Table 2. Laboratory type and value configuration

<table>
<thead>
<tr>
<th>Laboratory title</th>
<th>Minimal allowed value</th>
<th>Maximal allowed value</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>White blood cell count</td>
<td>3</td>
<td>12</td>
<td>k/cumm</td>
</tr>
<tr>
<td>Red blood cell count</td>
<td>3</td>
<td>7</td>
<td>m/cumm</td>
</tr>
<tr>
<td>Hemoglobin</td>
<td>10</td>
<td>19</td>
<td>gm/dl</td>
</tr>
<tr>
<td>Hematocrit</td>
<td>30</td>
<td>55</td>
<td>%</td>
</tr>
<tr>
<td>Mean corpuscular volume</td>
<td>70</td>
<td>100</td>
<td>fl</td>
</tr>
<tr>
<td>Mean cell hemoglobin</td>
<td>22</td>
<td>40</td>
<td>pg</td>
</tr>
<tr>
<td>Test</td>
<td>Lower Limit</td>
<td>Upper Limit</td>
<td>Unit</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>-------------</td>
<td>-------------</td>
<td>--------</td>
</tr>
<tr>
<td>Mean corpuscular hemoglobin concentration</td>
<td>28</td>
<td>40</td>
<td>g/dl</td>
</tr>
<tr>
<td>Red cell distribution width</td>
<td>9</td>
<td>17</td>
<td>%</td>
</tr>
<tr>
<td>Platelet count</td>
<td>120</td>
<td>450</td>
<td>k/cumm</td>
</tr>
<tr>
<td>Absolute neutrophils</td>
<td>60</td>
<td>80</td>
<td>%</td>
</tr>
<tr>
<td>Absolute Lymphocytes</td>
<td>15</td>
<td>35</td>
<td>%</td>
</tr>
<tr>
<td>Neutrophils</td>
<td>1</td>
<td>11</td>
<td>k/cumm</td>
</tr>
<tr>
<td>Lymphocytes</td>
<td>0.5</td>
<td>5</td>
<td>k/cumm</td>
</tr>
<tr>
<td>Monocytes</td>
<td>0.08</td>
<td>1.2</td>
<td>k/cumm</td>
</tr>
<tr>
<td>Eosinophils</td>
<td>0.08</td>
<td>0.6</td>
<td>k/cumm</td>
</tr>
<tr>
<td>Basophils</td>
<td>0</td>
<td>0.22</td>
<td>k/cumm</td>
</tr>
<tr>
<td>Sodium</td>
<td>125</td>
<td>155</td>
<td>mmol/L</td>
</tr>
<tr>
<td>Potassium</td>
<td>3</td>
<td>6</td>
<td>mmol/L</td>
</tr>
<tr>
<td>Chloride</td>
<td>90</td>
<td>115</td>
<td>mmol/L</td>
</tr>
<tr>
<td>Carbon dioxide</td>
<td>18</td>
<td>36</td>
<td>mmol/L</td>
</tr>
<tr>
<td>Anion gap</td>
<td>3</td>
<td>18</td>
<td>mmol/L</td>
</tr>
<tr>
<td>Glucose</td>
<td>60</td>
<td>140</td>
<td>mg/dL</td>
</tr>
<tr>
<td>Blood urea nitrogen</td>
<td>5</td>
<td>30</td>
<td>mg/dL</td>
</tr>
<tr>
<td>Creatinine</td>
<td>0.5</td>
<td>1.2</td>
<td>mg/dL</td>
</tr>
<tr>
<td>Total protein</td>
<td>5</td>
<td>10</td>
<td>gm/dL</td>
</tr>
<tr>
<td>Albumin</td>
<td>2.5</td>
<td>6</td>
<td>gm/dL</td>
</tr>
<tr>
<td>Calcium</td>
<td>7</td>
<td>12</td>
<td>mg/dL</td>
</tr>
<tr>
<td>Total bilirubin</td>
<td>0</td>
<td>1.2</td>
<td>mg/dL</td>
</tr>
<tr>
<td>Aspartate aminotransferase</td>
<td>12</td>
<td>42</td>
<td>U/L</td>
</tr>
<tr>
<td>Serum glutamic pyruvic transaminase</td>
<td>15</td>
<td>75</td>
<td>U/L</td>
</tr>
<tr>
<td>Alkaline phosphatase</td>
<td>40</td>
<td>150</td>
<td>U/L</td>
</tr>
<tr>
<td>Urine specific gravity</td>
<td>1.014</td>
<td>1.028</td>
<td>no unit</td>
</tr>
<tr>
<td>Urine pH</td>
<td>4.5</td>
<td>7.5</td>
<td>no unit</td>
</tr>
<tr>
<td>Urine red blood cells</td>
<td>0</td>
<td>3.5</td>
<td>rbc/hpf</td>
</tr>
<tr>
<td>Urine white blood cells</td>
<td>0</td>
<td>6</td>
<td>wbc/hpf</td>
</tr>
</tbody>
</table>
The FairGRecs Dataset: A Dataset for Producing Health-related Recommendations

Maria Stratigi¹, Haridimos Kondylakis², and Kostas Stefanidis³

¹ University of Tampere, Tampere, Finland, Maria.Stratigi@uta.fi
² ICS-FORTH, Heraklion, Greece, kondylak@ics.forth.gr
³ University of Tampere, Tampere, Finland, kostas.stefanidis@uta.fi

Abstract. Nowadays, the number of people who search for information related to health online has significantly increased, while the time of health professionals for recommending useful sources online has been reduced to a great extend. As such, providing valuable information to users for health related issues, based on their personal health profiles, in the form of suggestions, approved by their caregivers, can significantly improve the opportunities that users have to inform themselves online about health problems and possible treatments. However, due to several legal and ethical constraints, personal health profiles usually are not accessible. In this paper, we present FairGRecs, a synthetic dataset that can be used for evaluating and benchmarking methods that produce recommendations related to health documents based on individual health records. Specifically, FairGRecs can create, via a fully parametrized API, synthetic patients profiles, containing the same characteristics that exist in a real medical database, including both information about health problems and also relevant documents.

1 Introduction

Medicine is undergoing a revolution that is transforming the nature of health-care from reactive to preventive. One of these challenges in this revolution is the problem of the quality and the amount of information that can be found online [1], since health information is one of the most frequently searched topics on the Web. Especially during the last decade, the number of users who look online for health and medical information has dramatically increased. However despite the increase in those numbers and the vast amount of information currently available online, it is very hard for a patient to accurately judge the relevance of some information to his/her own case and to identify the quality of the provided information.

Furthermore, the optimal solution for patients is to be guided by healthcare providers to appropriate sources of information [1], [13]. Delivering accurate information sources to a patient, increases his/her knowledge and changes the way of thinking, which is usually referred as patient empowerment [7], [8]. As a result, the patient’s dependency for information from the doctor is reduced. Also, patients feel autonomous and more confident about the management of their disease [16]. To this direction, health providers have the history of their patient’s and their interests, in order to make an informed decision about the information that would likely be beneficial for them. However, health
providers have less and less time to devote to their patients. As such, guiding each individual patient appropriately is a really difficult task.

On the other hand, the use of group-dynamics-based principles of behavior change have been shown to be highly effective in enhancing social support through promoting group cohesion in physical activity [2], in reducing smoking relapse [3] and in promoting healthy dietary habits [10]. In small groups, therapy sessions enjoy a social component as participants can share experiences and discussion. In those therapy sessions, a caregiver can guide patients to more optimal resources over the Web. However, if identifying online information content for a single patient is a difficult task, identifying information for a group of participants is a really challenging one.

To this direction, in our work [14, 15], we focused on recommending interesting health documents, to groups of users. Our motivation was to offer a list of recommendations to a caregiver who is responsible for a group of patients. The recommended documents need to be relevant, based on the patients current profile, namely by exploiting the patients personal healthcare record (PHR) data. However, although it is really common for patients to look for health information and sometimes to rate related documents on the Web, their profiles are usually not accessible, neither linked to those documents. Among others, legal and ethical constraints prohibit the collection and the exploitation of such a dataset.

This way, in this paper, we present a synthetic dataset, FairGRecs, that can be used for evaluating and benchmarking methods that produce recommendations related to health documents. More specifically, we rely on the EMRBots dataset[4], which contains synthetic patients profiles, containing the same characteristics that exist in a real medical database, such as patients admission details, demographics, socioeconomic details, labs and medications, extending it with a document corpus and a rating dataset. By exploiting the FairGRecs dataset, interested users can create patients that have provided rankings for health documents. To link document contents with patients, we use the ICD10[5] ontology, namely the International Statistical Classification of Diseases and Related Health Problems, which is a standard medical classification list maintained by the World Health Organization. FairGRecs is fully parametrized and is offered via an API.[6]

This dataset has been used already for optimizing the Personal Health Information Recommender (PHIR) [5], [8], [9], developed within the EU project iManageCancer [6]. PHIR is a recommendation engine for recommending high quality cancer documents selected by health providers to patients.

The rest of this paper is structured as follows. Section 2 introduces our synthetic dataset, consisting of patients data, a document corpus and a ratings dataset. We also include a dataset creation example to showcase how to build these datasets. Section 3 describes the developed application programming interface, while Section 4 concludes with a discussion on the usefulness of the produced dataset.

---

[6] https://bitbucket.org/MariaStratigi/fairgrecs-dataset/overview
2 The FairGRecs Dataset

In general, a recommender system requires access to a certain amount of information – set of medicinal documents, a dataset containing ratings that the users have given to those documents and the personal health information of the users, to make suggestions to users. The absence of real data, motivated us to develop a fully parameterized tool that automatically creates the necessary datasets, that we do not have access to.

The first major obstacle is the acquisition of the dataset containing the personal health information of the users. Among others, legal and ethical constraints prohibit the collection and the exploitation of such a dataset. A further constraint is that the health information dataset has to be linked to a document corpus via a users’ ratings dataset. So the problem we face is the need of three interlinked datasets, that are void of any legal and ethical constraints. The first step to overcome this obstacle, is the adoption of the 10,000 chimeric patient profiles provided by EMRBots.

2.1 Patient Profiles Dataset

Unlike other methods that typically obscure or shift real patients’ data elements, the EMRBots proposed methodology is invulnerable in terms of security, because it does not rely on real data elements pulled from an existing Electronic Medical Record (EMR); therefore, it is not associated with privacy concerns in regard to individuals’ sensitive data.

The data is generated according to pre-defined criteria and is not based on any human data. These criteria are divided into Population-level and Patient-level characteristics. The first group offers an array of values in order to define demographics, such as gender, marital status, major language, ethnicity, date of birth and income level. After the completion of the Population-level configuration, there are \( n \) patients generated. For each such patient, the Patient-level configuration associates with them additional details. These details include randomly generated length of stay (in days) and start and end dates. Furthermore, each admission is associated with laboratory measurements and a chief complaint randomly selected from a list of International Classification of Diseases, Tenth Revision, Clinical Modification (ICD-10-CM) codes. In the last step of the patient-level configuration, laboratory values are added. Laboratory values are based on 35 common types - for instance, sodium levels, creatinine levels, or platelet count.

2.2 Document Corpus & User Preferences Dataset

One of the most important information a recommender system requires is a dataset containing the users preferences towards a set of items [12]. These preferences can take many formats, such as ratings, check-ins or textual reviews [11]. For this work, our chosen format is that of a ratings dataset. Most specifically, we produce a dataset containing ratings associated with users for particular health related documents.

Based on the patients’ health profiles dataset that we have already acquired, we generate two new datasets. The first is a document corpus that consist of documents’ id codes and their corresponding keywords, and the second is the ratings dataset, that incorporates the ratings given by the patients to the documents.
The generation process for both of these datasets, is fully parameterized. It is worth mentioning that, because our main focus is the development of the ratings dataset, heavy emphasis is given to the health profile of the patients. This is mostly presented as the health problems of each patient. As mentioned in the previous section this information is documented using ICD-10 codes.

**Document Corpus.** The generation of the document corpus, includes a document id and the corresponding keywords for each created document. As such we are not generating full text documents. To achieve that, we take advantage of the ICD-10 ontology tree. We generated a $numDocs$ number of documents, for each second level category of the ICD-10 ontology (i.e., for each node that belongs in the second level of the ontology tree). We will call these *primary nodes*. The ICD-10 ontology tree has 295 such nodes. Here, we make the assumption that a document cannot be related to more than one category. For example, a document cannot impart information for pregnancy and the perinatal period, cause these areas are represented by two different *primary nodes* in the tree.

For the documents’ corresponding keywords, we randomly selected $numKeyWords$ words from the description text of the nodes in each subsequent subtree of the *primary node* that the document belongs. From those descriptions, we removed the most common words – often called stopwords, such as “the”, “and”, “it” and made our selection from the rest. A visual description is provided in Figure 1.

![Diagram](image)

**Fig. 1.** The created documents will be anchored to the primary nodes, while their keywords will be selected from the subtree.

When ranking items based on human preferences, the most common observed distribution of these ratings is the power law distribution [4]. In order to make our dataset as plausible as possible, we depict this distribution by randomly selecting a $popularDocs$ number of documents that will be the most popular. These documents span through all the categories and are selected arbitrarily.
Ratings dataset. In order to create the ratings dataset, first we assume that all the patients have given at least a minimum non zero number of ratings. This assumption was made in order to make our datasets easier to work within the domain of recommender systems, and more specifically the systems that operate under a collaborative filtering design. In general these systems operate by finding similar users to a given user, and extrapolating an item’s rating based on the scores given by those users [14, 15]. If a user has not given any ratings, we will stumble upon the cold start problem, where we cannot find any similar users to him/her and subsequently, we will not be able to provide him/her with any recommendations.

In order to avoid this problem, we have surmised that all patients have given a numRatings number of ratings. Specifically, we have divided the patients into three groups – occasional, regular and dedicated. The users in each group have given low, average and high number of ratings, respectively. The number of ratings are randomly selected from a numerical range, based on which of the three groups the patient belongs.

Based on that number, we will create a corresponding number of rating nodes for each user. A rating node links a user to a document, i.e., the user has rated that document. A user cannot have more than one rating node for the same document, but can have many nodes for different documents that belong in the same category.

We have divided the user’s rating nodes into two groups; healthRelevant and nonRelevant. Using the health problems data (noted in ICD-10 nodes) of each user, the first group of ratings will go to documents belonging to the same subtree as one of their health problems, while the second group will be randomly assigned to the rest of the documents. Our assumption here is that the patients will be interested not only in documents regarding their health problems, but also to some extent in others as well.

Finally, in the last step, we assign rating values for each rating node that we generated previously. We choose the nodes randomly, and assign to them a value in the range of 1 to 5. The user is able to define the total number of ratings with a specific value that will be present in the rating dataset. This is accomplished as shown before with the use of percentages. For example, the administrator can define that the 25% of all ratings will have the value of 1.

2.3 Datasets Creation Example

In Section 2.2, we analyzed the proposed method of creating two datasets - documents and ratings - in lieu of real data. In Tables 1 and 2, we present the parameters needed for creating an example dataset and we briefly explain the values given to them. Furthermore, we selected to use the 10.000 patients chimeric dataset provided by the EMR-Bots. After all the necessary steps were completed the number of items in the document corpus was 79,650 and the total number of ratings generated was 1,576,872.

Figure 2 depicts the distribution of ratings in the documents. We partition the ratings in groups of 50. Most of the documents (71%) have received ratings in the range of [50-100]. In the second place (21%), we have the documents that have been rated from 0 to 50 times, while if we accumulate all the documents which have been rated more than 200 times, they merely make up of the 1.12% of the corpus. As expected, these results simulate a power law, where the prominent items are few, and the plethora of documents have very low popularity.
Table 1. The parameters needed to creating the document corpus.

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Explanation</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>numDocs</td>
<td>The number of documents created for each different category of health problems, based on the ICD10 ontology tree.</td>
<td>270</td>
</tr>
<tr>
<td>numKeyWords</td>
<td>The number of randomly selected keywords, attached to each document.</td>
<td>10</td>
</tr>
<tr>
<td>popularDocs</td>
<td>The number of documents, that will be most popular in each category, in order to simulate a power law distribution.</td>
<td>70</td>
</tr>
</tbody>
</table>

Table 2. The parameters needed to create the ratings dataset.

<table>
<thead>
<tr>
<th>Partitions</th>
<th>Parameter Name</th>
<th>Explanation</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Groups</td>
<td>Group occasional</td>
<td>Users give [20,100] ratings</td>
<td>50% of all patients</td>
</tr>
<tr>
<td></td>
<td>Group regular</td>
<td>Users give [100,250] ratings</td>
<td>30% of all patients</td>
</tr>
<tr>
<td></td>
<td>Group dedicated</td>
<td>Users give [250,500] ratings</td>
<td>20% of all patients</td>
</tr>
<tr>
<td>Scores</td>
<td>One</td>
<td>Ratings valued as 1</td>
<td>20% of all ratings</td>
</tr>
<tr>
<td></td>
<td>Two</td>
<td>Ratings valued as 2</td>
<td>10% of all ratings</td>
</tr>
<tr>
<td></td>
<td>Three</td>
<td>Ratings valued as 3</td>
<td>30% of all ratings</td>
</tr>
<tr>
<td></td>
<td>Four</td>
<td>Ratings valued as 4</td>
<td>20% of all ratings</td>
</tr>
<tr>
<td></td>
<td>Five</td>
<td>Ratings valued as 5</td>
<td>20% of all ratings</td>
</tr>
<tr>
<td>Ratings</td>
<td>healthRelevant</td>
<td>Ratings relevant to health problems</td>
<td>20% of user’s ratings</td>
</tr>
<tr>
<td></td>
<td>nonRelevant</td>
<td>Ratings not relevant to health problems</td>
<td>80% of user’s ratings</td>
</tr>
</tbody>
</table>
3 Application & Programming Interface

In order to streamline the creation process of the datasets and make it more user friendly, we developed an application and the corresponding Application Programming Interface (API). Those incorporate all the functions described in the previous section. They were both developed using Java, and they are available online \(^7\). The application interface, shown in Figure 3, is divided into four main tabs, each dedicated to a different part of the datasets creation process.

The first tab called ‘File Paths’, is a form for loading the two EMRbots dataset files, that contain the patients basic information and their health problems. In addition, the ICD-10 ontology is needed in a xml format, as well as a stopwords file containing the most common words in the English language.

The second tab is about the document corpus. The user needs to enter a numerical value, regarding the number of documents that will be created per category (i.e., first level nodes), and the number of keywords each document will have. The final input concerns the number of documents that will be popular per category, in order to simulate a power law distribution.

In the third tab, we have accumulated all the parameterized variables regarding the patients. These predominantly concern the patients partitioning into groups. Specifically, the user can divide the patients into the three groups by setting the percentage of the patients that belong in each group. The user is also able to define the minimum and maximum number of ratings per different group. Finally, the distribution of the user’s

\(^7\) https://bitbucket.org/MariaStratigi/fairgrecs-dataset/overview
ratings to documents that are relevant to his/her health problems is again accomplished by the use of percentages.

The last tab concerns the distribution of values to the ratings nodes. As before, the user defines the percentage of ratings with a specific value. The selection of a document to assign a value is done randomly.

4 Discussion & Conclusion

To the best of our knowledge, the FairGRecs dataset, is currently the only dataset available, combining personal health record, documents and ratings, offering a unique opportunity for experimenting with recommender systems in the health domain. As such it fills an important gap in the area of health recommender systems and it reuses and significantly extends state of the art datasets. Furthermore, is of particular interest to the Semantic Web Community, as in its core a widely used taxonomy is used, opening many possibilities for subsequent exploitation through reasoning techniques.

In addition, there is significant evidence of usage by the community of health recommender systems, as currently more and more platforms emerge enabling patients
to access high quality health related information. Nevertheless, we do not only offer a specific dataset but also an application and the corresponding API, enabling experimentation with endless possibilities, as well as its wider adoption and extensibility. The source code is also available allowing further extensions by the community.

References


Statistical Modeling of Clinical Data

Anoop Mayampurath | amayampurath@peds.bsd.uchicago.edu
Julie Johnson | jjohnso3@bsd.uchicago.edu

February 1, 2018
Agenda

- Objective
- Constructing your study
- Composition of clinical data
- How to perform descriptive analyses?
- How to perform complex analyses?
This workshop is about analyzing clinical data.
Types of healthcare data

- Claims data: patient demographics, diagnosis codes, dates of service, cost of service, etc.

- **EHR data: everything above plus vitals, labs, meds, interventions, reports, and notes.**

- Socioeconomic data: average income, crime, access to healthy food, pharmacies

- Self-reported data: personalized data, wearable technology
General tips

- Don’t be scared of messy data
- Understand probability
- Learn how to do data processing
- Learn how to do data modeling
- Interpretation over blind application
I think there are three main steps in a data science project: you *collect* data (and questions), *analyze* it (using visualization and models), then *communicate* the results. It's rare to walk this process in one direction: often your analysis will reveal that you need new or different data, or when presenting results you'll discover a flaw in your model.

- Hadley Wickham
Ask an interesting question.

Get the data.

Explore the data.

Model the data.

Communicate and visualize the results.

What is the scientific goal? What would you do if you had all the data? What do you want to predict or estimate?

How were the data sampled? Which data are relevant? Are there privacy issues?

Plot the data. Are there anomalies? Are there patterns?

Build a model. Fit the model. Validate the model.

What did we learn? Do the results make sense? Can we tell a story?

CS109: Harvard - Source:
General study designs

At the beginning of a study, there are several choices available to the researcher on how to conduct the study. These include:

- Descriptive (e.g. surveys, case studies)
- Associative (e.g. observational studies of type: outcome ~ exposure)
- Predictive (e.g. risk prediction)
- Review (e.g. literature review)
- Experimental (e.g. Randomized-Controlled Trials)
- Meta-analysis (i.e. combining the results of multiple studies)
Observational Study Design

Common types of observational study designs
- Case-Control study
- Retrospective cohort study
- Prospective cohort study
- Cross-sectional study
Case-Control study

Objective is to estimate the relative risk for an outcome from a specific variable (or risk factor or exposure) using odds ratios.

The dataset is built after the outcome is identified, following which the occurrence of previous exposure is determined.

Exposure is loosely defined as whether the variable of interest holds true or not.
Key things for a case-control study

- Controls must be comparable to case except without the occurrence of outcome.

- Non-matched case-control study: while building control, we ignore the number as well characteristics of the case.

- Matched case-control study: while building control, we take into account some characteristic of the case (e.g. gender). Ways to match are 1:n case-control matching, distribution-based matching, and propensity matching.

- Essentially, when matching, you are minimizing the effect of confounders.

- You can’t measure incidence rates (i.e. rate of outcome) in case-control studies.
Cohort Study

The idea is that you recruit subjects purely based on exposure status, i.e. none of them have developed an outcome. Then, you follow them in time until some of them develop an outcome. In other words, these are longitudinal studies.

- Prospective cohort study: Identify the study population at the beginning. Determine exposure status. Follow them through time.

- Retrospective cohort study: Historical record is collected for all exposed/non-exposed subjects. Determine current outcome status.

- Association is measured in terms of relative risk or using survival analysis
Differences

- Case-control study does not use an entire cohort. As a result, you cannot measure outcome rates accurately. Retrospective cohorts use the entire cohort.

- Sample size for case-control is dependent of rates of exposure, not outcome. The reverse is true for cohort studies, i.e., sample size is based on rates of outcome, not exposure.
How to choose?

- Cohort studies provide the best information about causality.

- In cohort studies (both prospective and retrospective), you can also measure associations with different outcomes for the same exposure.

- Prospective cohort studies while robust and controllable, are expensive to conduct, with a potential danger of patients dropping off from the study.

- Generally speaking, cohort studies work well with rare exposures. It does not work for rare outcomes, the sample size has to be very high for finding proper risk for incidence.
- Case control studies are simpler to conduct. They are quick and inexpensive and good for studying outbreaks.

- Case-control studies are more prone to bias and are less capable at showing a causal relationship.

- Case-control studies work well with rare outcomes, since you choose the outcome yourself.
Structure of clinical data

- Discrete
  - demographics, vitals, cultures, etc

- Narrative
  - admission notes, progress notes, discharge summaries, etc.

- Images
  - Xray, MRI, etc.
Analysis

- Identify outcome, exposure, and potential confounders.
- Perform an unadjusted analysis (Table 1 and Figure 1).
- Perform a fully-adjusted analysis.
Dataset

We have fake patient data (downloaded from EMRBots.org) to illustrate an example of clinical workflow.

- patient_demo.txt
  - [PatientGender] - Male/Female.
  - [PatientDateOfBirth] - Date Of Birth.
- **patient_encounter.txt**
  - `[Encounter_ID]` - an admission ID for the patient.
  - `[AdmissionStartDate]` - start date of encounter.
  - `[AdmissionEndDate]` - end date of encounter.

- **patient_diagnosis.txt**
  - `[Encounter_ID]` - an admission ID for the patient.
  - `[PrimaryDiagnosisCode]` - ICD10 code for admission’s primary diagnosis.
  - `[PrimaryDiagnosisDescription]` - admission’s primary diagnosis description.
- **patients_labs.Rdata**
  
  - `[Encounter_ID]` - an admission ID for the patient.
  - `[LabName]` - lab’s name
  - `[LabValue]` - lab’s value
  - `[LabUnits]` - lab’s units.
  - `[LabDateTime]` - date.
Study goal: Identify the risk factors for malignant neoplasm*


We will also look at the association between patient characteristics and diagnosis of the disease.
**Task 1: Getting the outcome.**

There are a few options that generally used to determine patient outcome.

- ICD9/10
- Medications/Interventions
- Other data sources
Let's set up our R environment

```r
rm(list = ls())
library(plyr)
library(dplyr)
library(lubridate)
```
Let’s read in our dataset into a data frame

d.dx <- read.csv("~/Google Drive/teaching/2018_CRI_Seminar/data/patient_diagnosis.csv")
names(d.dx)

# [1] "Patient_ID"          "Encounter_ID"
# [3] "PrimaryDiagnosisCode" "PrimaryDiagnosisDescription"
Most pre-processing can be done using the following commands

- **mutate()**: To create new variables based on some operation of old variables.
- **filter()**: To subset a set of rows based on values of a variable.
- **select()**: To select variables. Also used to remove variables.
- **merge()**: To combine data frames using common variables.

**KEY point**: All four use data frames as both input (the first argument) and output.
Using `filter()`

What if we wanted to look identify patients diagnosed with malignant neoplasm?

Let’s choose that subset using the `filter()` command.

**KEY:** The `filter()` function is used to “subset” data, i.e. selecting rows according to a particular condition. In this example, we want to subset `d.dx` by selecting patients (i.e. the rows) who had malignant neoplasm (i.e. the condition).

The general syntax is

```
data_frame_new <- filter(data_frame_old, condition)
```
# get all patients with malignant neoplasm

d.mp <- d.dx %>%
  filter(grepl('Malignant neoplasm', PrimaryDiagnosisDescription))

head(d.mp)

## Patient_ID Encounter_ID PrimaryDiagnosisCode
## 1     101009     101009_2                C34.1
## 2     101009     101009_3                  C67
## 3     101407     101407_2                  C33
## 4     101407     101407_5                  C47
## 5     101407     101407_1                C63.0
## 6     105700     105700_1                C14.8

## PrimaryDiagnosisDescription
## 1 Malignant neoplasm of upper lobe, bronchus or lung
## 2 Malignant neoplasm of bladder
## 3 Malignant neoplasm of trachea
## 4 Malignant neoplasm of peripheral nerves and autonomic nervous system
## 5 Malignant neoplasm of epididymis
## 6 Malignant neoplasm of overlapping sites of lip, oral cavity and pharynx
**Using pipe**

A good way to do combine successive operations using data frames is to use the `>%` symbol. Why? Instead of writing multiple lines, you can achieve the same result using single line through the pipe (">%") operator.

The general syntax is: `output = data_frame %>% operation_1 %>% operation_2 %>% operation_3..`
Using `select()`

If you want to choose columns into another data frame, you can use the `select()` function.

KEY: The `select()` function is used to choose (or remove) columns of choice. Once again, `select()` (like `filter()`) works with data frames. The general syntax is -

```r
data_frame_new <- select(data_frame_old,
                        c(col1, col2, etc))
```
d.mp_ids <- d.mp %>%
  select(Patient_ID, Encounter_ID) %>% unique()
head(d.mp_ids)

<table>
<thead>
<tr>
<th>#</th>
<th>Patient_ID</th>
<th>Encounter_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>101009</td>
<td>101009_2</td>
</tr>
<tr>
<td>2</td>
<td>101009</td>
<td>101009_3</td>
</tr>
<tr>
<td>3</td>
<td>101407</td>
<td>101407_2</td>
</tr>
<tr>
<td>4</td>
<td>101407</td>
<td>101407_5</td>
</tr>
<tr>
<td>5</td>
<td>101407</td>
<td>101407_1</td>
</tr>
<tr>
<td>6</td>
<td>105700</td>
<td>105700_1</td>
</tr>
</tbody>
</table>

Note: If you put a - in front of the variable, (i.e. say -c(Patient_ID)), you will REMOVE/DE-SELECT these columns.
There is a difference between patients and patient admissions.

```r
cat("Number of admissions with malignant neoplasm",
    d.mp_ids %>% select(Encounter_ID) %>% unique() %>% nrow(), "\n")
```

## Number of admissions with malignant neoplasm 4375

```r
cat("Number of patients with malignant neoplasm",
    d.mp_ids %>% select(Patient_ID) %>% unique() %>% nrow(), "\n")
```

## Number of patients with malignant neoplasm 3589
d.no_mp_ids <- d.dx %>%
  filter(!(Encounter_ID %in% unlist(d.mp_ids$Encounter_ID))) %>%
  select(Patient_ID, Encounter_ID) %>%
  unique()

cat("Number of admissions without malignant neoplasm",
    d.no_mp_ids %>% select(Encounter_ID) %>% unique() %>% nrow(), "\n")

## Number of admissions without malignant neoplasm 31768

## Number of admissions without malignant neoplasm 31768
d.mp_ids$outcome <- 1
d.no_mp_ids$outcome <- 0
d.mp_outcome <- rbind(d.mp_ids, d.no_mp_ids)
write.csv(d.mp_outcome,
    "~/Google Drive/teaching/2018_CRI_Seminar/results/outcome/mp_outcome.csv",
    row.names = FALSE)
Task 2: Put together a descriptive analysis.

Let’s begin by putting together clinical characteristics (e.g. age, gender, race, LOS) for each admission.

We will need

- encounter ids of interest
- Age: date of birth and admission start date
- LOS: admissions start and end dates.
Using `merge()`

`merge()` is used to combine two datasets based on variables (keys)

Here is a great cheat-sheet for understanding `merge()` in a greater detail:
http://stat545.com/bit001_dplyr-cheatsheet.html

d.enc_info <- read.table("~/Google Drive/teaching/2018_CRI_Seminar/data/patient_encounter.txt",
                       sep = "$\t$", header = TRUE)
d.mp_outcome <- read.csv("~/Google Drive/teaching/2018_CRI_Seminar/results/outcome/mp_outcome.csv")
d.demo <- read.table("~/Google Drive/teaching/2018_CRI_Seminar/data/patient_demo.txt",
                      sep = "$\t$", header = TRUE)

d.cohort <- merge(d.mp_outcome, d.enc_info, by = c("Encounter_ID", "Patient_ID"))
d.cohort <- merge(d.cohort, d.demo, by = c("Patient_ID"))
Using `mutate()`

`mutate()` is used for creating new variables using a combination of existing variables.

The general syntax is: `data_frame_new <- mutate(data_frame_old, 
new_column1 = do_stuff(old_column1), 
new_column2 = doStuff(old_column2))`
Handling date and time

We will use the lubridate() package for this purpose. More details and examples can be found at [https://cran.r-project.org/web/packages/lubridate/vignettes/lubridate.html](https://cran.r-project.org/web/packages/lubridate/vignettes/lubridate.html)

```r
Sys.setenv(tz = "America/Chicago")

d.cohort <- d.cohort %>%
  mutate(AdmissionStartDate = ymd_hms(AdmissionStartDateTime),
         AdmissionEndDate = ymd_hms(AdmissionEndDateTime),
         PatientDateOfBirth = ymd_hms(PatientDateOfBirth))
```
We used `ymd_hms` because the format in this dataset was `YYYY-MM-YY HH:MM:SS`.

If the format was `dd-mm-yy`, you would use `dmy()`. `lubridate()` can identify a variety of separators between the date-time components.
Calculating age and LOS.

```r
d.cohort <- d.cohort %>%
  mutate(PatientAge = interval(PatientDateOfBirth, AdmissionStartDate) / dyears(1))

d.cohort <- d.cohort %>%
  mutate(LOS = interval(AdmissionStartDate, AdmissionEndDate) / ddays(1))
```
Some take aways

In case you hadn’t noticed, the dataset was in a format that was ready to analyze. Notably,

- Every variable was in a separate column with readable column names
- Every observation was in a separate row
- The data frame (generally speaking) contained variables that are consistent with a particular theme. For e.g, patient demographics is different from patient vitals
- The data frame had at least one unique identifier from which it possible to link different tables
The importance of summarizing

Really, you are looking to test the quality of your dataset

- Missing values
- Extreme values
- Consistent units
- Remove things that shouldn’t be there in the first place
- NOTE: 80% of your analyses will be prepping the data. dplyr() makes it much easier to do so

Check for consistency: continuous variables

For continuous variables, use the quantile() function to check for outliers. The quantile() function will return the 25%, 50%, and 75% quantiles along with max and min. Use ?quantile to study it further.

```r
quantile(d.cohort$PatientAge)
```

```r
##       0%      25%      50%      75%     100%
## 18.01184 25.83876 38.58597 54.39640 92.95689
```
Check for consistency: categorical variables

For categorical variables, use the `summary()` function for counting the number of entries corresponding to a particular categorical level.

```
summary(d.cohort$PatientRace)
```

```r
##  # African American Asian Unknown White
##     5403     8284     4701   17755
```

```
summary(as.factor(d.cohort$outcome))
```

```r
##  # 0 1
## 31768 4375
```
We can start constructing our Table 1.

<table>
<thead>
<tr>
<th>#</th>
<th>Column</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Patient_ID</td>
</tr>
<tr>
<td>4</td>
<td>AdmissionStartDate</td>
</tr>
<tr>
<td>7</td>
<td>PatientDateOfBirth</td>
</tr>
<tr>
<td>10</td>
<td>LOS</td>
</tr>
<tr>
<td>2</td>
<td>Encounter_ID</td>
</tr>
<tr>
<td>5</td>
<td>AdmissionEndDate</td>
</tr>
<tr>
<td>8</td>
<td>PatientRace</td>
</tr>
<tr>
<td>3</td>
<td>outcome</td>
</tr>
<tr>
<td>6</td>
<td>PatientGender</td>
</tr>
<tr>
<td>9</td>
<td>PatientAge</td>
</tr>
<tr>
<td>Variable</td>
<td>Patient admissions with outcome (n=4,375)</td>
</tr>
<tr>
<td>--------------------------</td>
<td>------------------------------------------</td>
</tr>
<tr>
<td>Age, mean(sd), yr</td>
<td></td>
</tr>
<tr>
<td>Gender</td>
<td>Male, n (%)</td>
</tr>
<tr>
<td></td>
<td>Female, n (%)</td>
</tr>
<tr>
<td>LOS, median (IQR)</td>
<td></td>
</tr>
</tbody>
</table>
Comparison of continuous variables

Let’s compare age between the two groups

```r
# Filter the data by outcome
d.1 <- filter(d.cohort, outcome == 1)
d.0 <- filter(d.cohort, outcome == 0)

# Calculate mean age for each outcome
cat("Mean age, outcome 1: ", mean(d.1$PatientAge), "\n")
## Mean age, outcome 1: 42.06632

cat("Mean age, outcome 0: ", mean(d.0$PatientAge), "\n")
## Mean age, outcome 0: 41.70317

# Calculate standard deviation for each outcome
cat("SD age, outcome 1: ", sd(d.1$PatientAge), "\n")
## SD age, outcome 1: 18.17401

cat("SD age, outcome 0: ", sd(d.0$PatientAge), "\n")
## SD age, outcome 1: 18.04217
```
```r
print(t.test(d.1$PatientAge, d.0$PatientAge))
```

```r
## Welch Two Sample t-test
## data:  d.1$PatientAge and d.0$PatientAge
## t = 1.2402, df = 5627.6, p-value = 0.215
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.2108925  0.9371819
## sample estimates:
## mean of x mean of y
##  42.06632  41.70317
```
Statistical inference has three important concepts

- the null hypothesis
- the alternate hypothesis
- the p-value
For comparing means between two populations

- null: there is no difference in average age between groups
- alternate: there is difference in average age between groups
- the smaller the p-value, the more confident in rejecting the null hypothesis
Step-wise logic

- Assume null is true
- If the data fails to contradict null beyond a reasonable doubt, null is not rejected
- Don’t assume null is true if we don’t reject it (tricky!)
- If not rejected, null is simply a possible explanation for data behavior
- Only when the data contradicts null strongly is the null rejected and the alternative accepted
Comparision of categorical variables

Let’s compare the gender variable with respect to our outcome.

First let’s build a 2x2 table.

gender.table <- with(d.cohort, table(outcome, PatientGender))
gender.table

```r
# PatientGender
# outcome Female  Male
# 0   16584 15184
# 1    2292  2083
```
Chi-squared testing Null: Outcome is not associated with gender

Alternative : Outcome is associated with gender

P-value: the smaller the p-value, the more confident in rejecting the null hypothesis
Looking at the result below, we can say that no association was observed between outcome and gender.

chisq.test(gender.table)

```r
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  gender.table
## X-squared = 0.045645, df = 1, p-value = 0.8308
```
Comparing medians

For variables that are not distributed normally (e.g. length of stay, which is skewed), we compare the median and the inter-quantile range (IQR). In R, we use median() and quantile() to get these values. Statistical comparison between groups is done by the Mood test (in R this is mood.test()). Note that comparing means in skewed distributions is also done using non-parametric tests such as Wilcoxon rank sum test (wilcox.test() in R). For further details, see https://www.r-bloggers.com/example-2014-6-comparing-medians-and-the-wilcoxon-rank-sum-test/.
Table 1

We now have everything we need to create table 1.

<table>
<thead>
<tr>
<th>#</th>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><code>Patient_ID</code></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td><code>Encounter_ID</code></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td><code>outcome</code></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td><code>AdmissionStartDate</code></td>
<td>Admission Start Date</td>
</tr>
<tr>
<td>5</td>
<td><code>AdmissionEndDate</code></td>
<td>Admission End Date</td>
</tr>
<tr>
<td>6</td>
<td><code>PatientGender</code></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td><code>PatientDateOfBirth</code></td>
<td>Patient Date of Birth</td>
</tr>
<tr>
<td>8</td>
<td><code>PatientRace</code></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td><code>PatientAge</code></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td><code>LOS</code></td>
<td></td>
</tr>
<tr>
<td>Variable</td>
<td>Patient admissions with outcome (n=4,375)</td>
<td>Patient admissions without outcome (n=31,768)</td>
</tr>
<tr>
<td>------------------------</td>
<td>------------------------------------------</td>
<td>---------------------------------------------</td>
</tr>
<tr>
<td>Age, mean(sd), yr</td>
<td>42 (18)</td>
<td>42 (18)</td>
</tr>
<tr>
<td>Gender</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Male, n (%)</td>
<td>2,083 (48)</td>
<td>15,184 (48)</td>
</tr>
<tr>
<td>Female, n (%)</td>
<td>2,292 (52)</td>
<td>16,584 (52)</td>
</tr>
<tr>
<td>LOS, median (IQR), days</td>
<td>11 (6,15)</td>
<td>11 (6, 16)</td>
</tr>
</tbody>
</table>
Task 3: Compiling the variables of interest.

We create a feature matrix that has the most-recent lab values associated with the encounter along with the outcome. This has already been compiled and given.

```r
# this will load up a data frame called d.enc_labs
load("~/Google Drive/teaching/2018_CRI_Seminar/results/features/mp_most_recent_labs.RData")

# Merge with outcome to get a feature matrix
d.features <- merge(d.cohort, d.enc_labs, by = c("Encounter_ID"))
```
# some cleaning
d.features <- d.features %>%
   select(-c(Encounter_ID, Patient_ID, AdmissionStartDate, AdmissionEndDate, PatientDateOfBirth, LabDateTime))

names(d.features)

## [1] "outcome"                     "PatientGender"
## [3] "PatientRace"                 "PatientAge"
## [5] "LOS"                          "CBC: ABSOLUTE NEUTROPHILS"
## [7] "CBC: HEMATOCRIT"              "CBC: HEMOGLOBIN"
## [9] "CBC: PLATELET COUNT"          "CBC: RED BLOOD CELL COUNT"
## [13] "METABOLIC: BILI TOTAL"        "METABOLIC: CALCIUM"
## [15] "METABOLIC: POTASSIUM"         "METABOLIC: SODIUM"
## [17] "URINALYSIS: PH"

# save work
save(list = c("d.features"),
     file = "~/Google Drive/teaching/2018_CRI_Seminar/results/features/mp_study_features.RData")
Task 4: Regression

In association studies, we want to understand the relationship between and exposure and outcome. We do this sequentially:

- outcome \sim exposure (called as unadjusted analysis)
- outcome \sim exposure + confounders (called as adjusted analysis)

The idea is to see if the relationship persists after adjustment of confounders.
The choice of linear or logistic regression depends on the outcome.

- if outcome is continuous, perform linear regression
- if outcome is binary, perform logistic regression
Linear Regression

The OLS mode for linear regression takes the form:

\[ Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p \]

We know that
- \( Y \) is the response/outcome
- all \( X_i \) are predictors/variables/features
- \( \beta \)'s are parameters/model coefficients/weights and are estimated using least squares
\( \beta_0 \) is the intercept, which is \( Y \) when all continuous predictors are 0 and all categorical predictors are set to reference.

For every unit increase in \( X_i \), the response \( Y \) changes by \( \beta_i \).
For example, consider modeling price of car (in $1000) against years from purchase (in years)

\[
    \text{car\_price} = \beta_0 + \beta_1 \text{years\_from\_purchase}
\]

Let \( \beta_0 = 10 \) and \( \beta_1 = -0.95 \).

The interpretation is

- at year of purchase the car price was $10,000.
- for every additional year, the car price will go down by $950.
Logistic Regression

The predictor $Y$ is binary (i.e. 0 and 1). Consider a simple model with response $Y$ and a single predictor $X$.

In logistic regression, we look at the conditional probability of $Y$ being 1 given $X$.

$$P(Y = 1 | X)$$
Odds

\[
Odds = \frac{\text{Probability of event}}{1 - \text{Probability of event}}
\]
Odds and probability are not the same.

Given a scenario where the mortality rate for an admitting patient is 20%, what are the odds that a patient will die?
Prob(Death) = 0.2

Odds(Death) = 0.2/0.8 = 1/4 = 0.25

For every patient who dies, there are four patients who will survive.
Consider the odds for $Y$ being 1 for a single variable model.

\[
Odds = \frac{P(Y = 1|X)}{1 - P(Y = 1|X)}
\]

In logistic regression,

\[
\log(Odds) = \beta_0 + \beta_1 X
\]
For multiple predictors,

\[
\log \frac{P(Y = 1|X)}{1 - P(Y = 1|X)} = \beta_0 + \beta_1 X_1 + \beta_2 X_2 \ldots \beta_p X_p
\]

If \( \beta_j > 0 \), then \( \exp(\beta_j) > 1 \), and the odds increase.

If \( \beta_j < 0 \), then \( \exp(\beta_j) < 1 \), and the odds decrease.

The p-value of \( \beta_i \) will indicate the significance of that coefficient.

In order to correctly interpret the model, you have to look at both the p-value and the OR.
Suppose we are interested in patient mortality from trauma patients who suffered out-of-hospital cardiac arrests.

What is the response?

What is the predictor?
Hypothetical example

Suppose we are interested in patient mortality from trauma patients who suffered out-of-hospital cardiac arrests.

What is the response?

- $Y$ is patient dying in the hospital ($1 = \text{Yes}, \ 0 = \text{No}$)

What is the predictor?

- $X$ whether a trauma patient had an out-of-hospital cardiac arrest ($1 = \text{Yes}, \ 0 = \text{No}$)
Hypothetical example

Suppose we are interested in patient mortality from trauma patients who suffered out-of-hospital cardiac arrests.

We perform a logistic regression, and we get $\beta_1 = 0.18$, p-value < 0.001

This means that
- (a) the log-odds of death increases by 0.18 when a patient comes in with out-of-hospital cardiac arrest
Hypothetical example

Suppose we are interested in in-hospital patient mortality from incoming trauma where patients suffer out-of-hospital cardiac arrests.

We perform a logistic regression, and we get $\beta_1 = 0.18$, p-value < 0.001,

Odds ratio = $\exp(0.18) = 1.20$ (95%CI: 1.14, 1.30)

This means that
- (a) the log-odds of death increases by 0.18 when a patient comes in with out-of-hospital cardiac arrest
- (b) the odds-ratio increases by 1.2 when a patient comes in with out-hospital cardiac arrest.
Hypothetical example

Suppose we are interested in in-hospital patient mortality from incoming trauma where patients suffer out-of-hospital cardiac arrests.

We perform a logistic regression, and we get $\beta_1 = 0.18$, $p < 0.001$

Odds ratio $= \exp(0.18) = 1.20$ (95% CI: 1.14, 1.30)

This means that for an incoming trauma that is from a out-of-hospital cardiac arrest, likelihood of patient dying in the hospital increases by 20%.
Hypothetical example

Suppose we are interested in in-hospital patient mortality from incoming trauma where patients suffer out-of-hospital cardiac arrests.

We perform a logistic regression, and we get $\beta_1 = 0.18$, and the p-value is not significant (i.e. > 0.001)

Odds ratio = $\exp(0.18) = 1.20$ (95% CI: 0.89, 1.70)

This means that no significant associations can be drawn from this study.
Task 4: Regression

```r
m1 <- glm(outcome ~ ., data = d.features, family = "binomial")
```
summary(m1)

```
##
## Call:
## glm(formula = outcome ~ ., family = "binomial", data = d.features)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.5766  -0.5177  -0.5043  -0.4892   2.1608
##
## Coefficients:
##                              Estimate Std. Error z value Pr(>|z|)
## (Intercept)                   -1.8703611  0.4187966  -4.466 7.97e-06 ***
## PatientGenderMale             -0.0117749  0.0325666  -0.362  0.71768
## PatientRaceAsian              0.0513524  0.0547992   0.937  0.34871
## PatientRaceUnknown            0.0355191  0.0625141   0.568  0.56991
## PatientRaceWhite              0.0759927  0.0487248   1.560  0.11885
## PatientAge                    -0.0009372  0.0008968  -1.045  0.29600
## LOS                            -0.0047828  0.0031642  -1.512  0.13065
## `CBC: ABSOLUTE NEUTROPHILS`   -0.0001219  0.0028179  -0.043  0.96548
## `CBC: HEMATOCRIT`             -0.0003363  0.0022647  -0.148  0.88196
## `CBC: HEMOGLOBIN`             -0.0061471  0.0062494  -0.984  0.32529
## `CBC: PLATELET COUNT`         0.0001796  0.0001704   1.054  0.29183
## `CBC: RED BLOOD CELL COUNT`   -0.0241488  0.0140585  -1.718  0.08585 .
## `CBC: WHITE BLOOD CELL COUNT` -0.0036096  0.0062556  -0.577  0.56393
## `METABOLIC: ALBUMIN`          -0.0118067  0.0160323  -0.736  0.46147
## `METABOLIC: BILI TOTAL`       0.0106724  0.0465493   0.229  0.81866
## `METABOLIC: CALCIUM`          0.0023245  0.0112481   0.207  0.83628
## `METABOLIC: POTASSIUM`        0.0062358  0.0187237   0.333  0.73910
## `METABOLIC: SODIUM`           -0.0017143  0.0018733  -0.915  0.36013
## `URINALYSIS: PH`             -0.0490157  0.0187973  -2.608  0.00912 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
```
Null deviance: 26266 on 35591 degrees of freedom
Residual deviance: 26246 on 35573 degrees of freedom
(551 observations deleted due to missingness)
AIC: 26284

Number of Fisher Scoring iterations: 4
<table>
<thead>
<tr>
<th>variable</th>
<th>p.value</th>
<th>OR</th>
<th>OR_2.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>7.968199e-06</td>
<td>0.1540680</td>
<td>0.06777298</td>
</tr>
<tr>
<td><code>CBC: ABSOLUTE NEUTROPHILS</code></td>
<td>9.654819e-01</td>
<td>0.9998781</td>
<td>0.99437070</td>
</tr>
<tr>
<td><code>CBC: HEMATOCRIT</code></td>
<td>8.819557e-01</td>
<td>0.9996638</td>
<td>0.99523621</td>
</tr>
<tr>
<td><code>CBC: HEMOGLOBIN</code></td>
<td>3.252950e-01</td>
<td>0.9938717</td>
<td>0.98177113</td>
</tr>
<tr>
<td><code>CBC: PLATELET COUNT</code></td>
<td>2.918346e-01</td>
<td>1.0001796</td>
<td>0.99984570</td>
</tr>
<tr>
<td><code>CBC: RED BLOOD CELL COUNT</code></td>
<td>8.584525e-02</td>
<td>0.9761404</td>
<td>0.94960526</td>
</tr>
<tr>
<td><code>CBC: WHITE BLOOD CELL COUNT</code></td>
<td>5.639268e-01</td>
<td>0.9963969</td>
<td>0.98425419</td>
</tr>
<tr>
<td><code>METABOLIC: ALBUMIN</code></td>
<td>4.614668e-01</td>
<td>0.9882627</td>
<td>0.95768775</td>
</tr>
<tr>
<td><code>METABOLIC: BILI TOTAL</code></td>
<td>8.186576e-01</td>
<td>1.0107296</td>
<td>0.92259621</td>
</tr>
<tr>
<td><code>METABOLIC: CALCIUM</code></td>
<td>8.362789e-01</td>
<td>1.0023272</td>
<td>0.98047136</td>
</tr>
<tr>
<td><code>METABOLIC: POTASSIUM</code></td>
<td>7.391031e-01</td>
<td>1.0062552</td>
<td>0.96999414</td>
</tr>
<tr>
<td><code>METABOLIC: SODIUM</code></td>
<td>3.601293e-01</td>
<td>0.9982872</td>
<td>0.99462812</td>
</tr>
<tr>
<td><code>URINALYSIS: PH</code></td>
<td>9.118144e-03</td>
<td>1.0502368</td>
<td>1.0122551</td>
</tr>
<tr>
<td>LOS</td>
<td>1.306524e-01</td>
<td>0.9952286</td>
<td>0.98907451</td>
</tr>
<tr>
<td>PatientAge</td>
<td>2.960048e-01</td>
<td>1.0009377</td>
<td>0.99917528</td>
</tr>
<tr>
<td>PatientGenderMale</td>
<td>7.176778e-01</td>
<td>0.9882942</td>
<td>0.92715513</td>
</tr>
<tr>
<td>PatientRaceAsian</td>
<td>3.487060e-01</td>
<td>1.0526938</td>
<td>0.94579455</td>
</tr>
<tr>
<td>PatientRaceUnknown</td>
<td>5.699140e-01</td>
<td>1.0361575</td>
<td>0.91655239</td>
</tr>
<tr>
<td>PatientRaceWhite</td>
<td>1.188467e-01</td>
<td>1.0789547</td>
<td>0.98130392</td>
</tr>
</tbody>
</table>
Things that could have gone wrong.

- We chose the most-recent lab for that encounter.
- We chose the lab values within the same encounter that was diagnosed with the condition.
- We chose patient admissions vs. patients.
Other topics

Visualization

Top 50 ggplot2 Visualizations - The Master List (With Full R Code)

What type of visualization to use for what sort of problem? This tutorial helps you choose the right type of chart for your specific objectives and how to implement it in R using ggplot2.

This is part 3 of a three part tutorial on ggplot2, an aesthetically pleasing (and very popular) graphics framework in R. This tutorial is primarily geared towards those having some basic knowledge of the R programming language and want to make complex and nice looking charts with R ggplot2.

- **Part 1**: Introduction to ggplot2, covers the basic knowledge about constructing simple ggpplots and modifying the components and aesthetics.
- **Part 2**: Customizing the Look and Feel, is about more advanced customization like manipulating legend, annotations, multiplots with faceting and custom layouts
- **Part 3**: Top 50 ggplot2 Visualizations - The Master List, applies what was learnt in part 1 and 2 to construct other types of ggpplots such as bar charts, boxplots etc.
Survival Analysis

- If you want to model time-to-event (such as death) on censored data.
- We want to model the probability that an observation can survive after a time point $t$.

- We calculate the hazard function, which is simply the probability that the event will occur in the next instant, given survival till time point $t$.

- Cox Proportional-Hazard model: estimate the effects of your variables/covariates on survival.

- use survival() package in R
Prediction

- Sensitivity/Specificity/Type I error
- Receiver Operating characteristic (ROC), Area under the Curve (AUC)
- Training/Testing/Cross-validation

- Machine learning models
  - Logistic Regression
  - Decision Trees/Random Forests
  - Support Vector Machines
  - Artificial neural network
  - Deep learning
Patient Subtyping via Time-Aware LSTM Networks

Inci M. Baytas
Computer Science and Engineering
Michigan State University
428 S Shaw Ln.
East Lansing, MI 48824
baytasin@msu.edu

Cao Xiao
Center for Computational Health
IBM T. J. Watson Research Center
1101 Kitchawan Rd
Yorktown Heights, NY 10598
cxiao@us.ibm.com

Xi Zhang
Healthcare Policy and Research
Weill Cornell Medical School
Cornell University
New York, NY 10065
sheryl.zhangxi@gmail.com

Fei Wang
Healthcare Policy and Research
Weill Cornell Medical School
Cornell University
New York, NY 10065
few2001@med.cornell.edu

Anil K. Jain
Computer Science and Engineering
Michigan State University
428 S Shaw Ln.
East Lansing, MI 48824
jain@cse.msu.edu

Jiayu Zhou
Computer Science and Engineering
Michigan State University
428 S Shaw Ln.
East Lansing, MI 48824
jiayuz@msu.edu

ABSTRACT
In the study of various diseases, heterogeneity among patients usually leads to different progression patterns and may require different types of therapeutic intervention. Therefore, it is important to study patient subtyping, which is grouping of patients into disease characterizing subtypes. Subtyping from complex patient data is challenging because of the information heterogeneity and temporal dynamics. Long-Short Term Memory (LSTM) has been successfully used in many domains for processing sequential data, and recently applied for analyzing longitudinal patient records. The LSTM units are designed to handle data with constant elapsed times between consecutive elements of a sequence. Given that time lapse between successive elements in patient records can vary from days to months, the design of traditional LSTM may lead to suboptimal performance. In this paper, we propose a novel LSTM unit called Time-Aware LSTM (T-LSTM) to handle irregular time intervals in longitudinal patient records. We learn a subspace decomposition of the cell memory which enables time decay to discount the memory content according to the elapsed time. We propose a patient subtyping model that leverages the proposed T-LSTM in an auto-encoder to learn a powerful single representation for sequential records of patients, which are then used to cluster patients into clinical subtypes. Experiments on synthetic and real world datasets show that the proposed T-LSTM architecture captures the underlying structures in the sequences with time irregularities.

KEYWORDS
Patient subtyping, Recurrent Neural Network, Long-Short Term Memory

1 INTRODUCTION
Clinical decision making often relies on medical history of patients. Physicians typically use available information from past patient visits such as lab tests, procedures, medications, and diagnoses to determine the right treatment. Furthermore, researchers use medical history and patient demographics to discover interesting patterns in patient cohorts, to study prognosis of different types of diseases, and to understand effects of drugs. In a nutshell, large-scale, systematic and longitudinal patient datasets play a key role in the healthcare domain. Examples such as Electronic Health Records (EHRs), whose adoption rate increased by 5% between 2014 and 2015 [1] in the healthcare systems in the United States, facilitate a systematic collection of temporal digital health information from variety of sources.

With the rapid development of computing technologies in healthcare, longitudinal patient data are now beginning to be readily available. However, it is challenging to analyze large-scale heterogeneous patient records to infer high level information embedded in patient cohorts. This challenge motivates the development of computational methods for biomedical informatics research [5, 17, 24, 28, 31]. These methods are required to answer different questions related to disease progression modeling and risk prediction [9, 10, 14, 22, 30, 32].

Patient Subtyping, which seeks patient groups with similar disease progression pathways, is crucial to address the heterogeneity in the patients which ultimately leads to precision medicine where patients are provided with treatments tailored to their unique health status. Patient subtyping facilitates the investigation of a particular type of complicated disease condition [5]. From the data mining...
Diagnoses
Visit 1
ICD-9:
• 42789
• 41401
• V861
• 4280
• ... V4511
• V1251
• V5861
• V4589
• 2875
Diagnoses
Visit 2
ICD-9:
• 2766
• 5856
• 40301
• 4254
• 28529
• 7100
• 78909

perspective, patient subtyping is posed as an unsupervised learning task of grouping patients according to their historical records. Since these records are longitudinal, it is important to capture the relationships and the dependencies between the elements of the record sequence in order to learn more effective and robust representations, which can then be used in the clustering stage to obtain the patient groups.

One powerful approach which can capture underlying structure in sequential data is Recurrent Neural Networks (RNNs), which have been applied to many areas such as speech recognition [16], text classification [21], video processing [13, 26], and natural language processing [27]. In principle, time dependencies between the elements can be successfully captured by RNNs, however traditional RNNs suffer from vanishing and exploding gradient problems. To handle these limitations, different variants of RNN have been proposed. Long-Short Term Memory (LSTM) [18] is one such popular variant which can handle long term event dependencies by utilizing a gated architecture. LSTM has recently been applied in health informatics [4, 6] with promising results.

One limitation of the standard LSTM networks is that it cannot deal with irregular time intervals. But, the time irregularity is common in many healthcare applications. To illustrate this, one can consider patient records, where the time interval between consecutive visits or admissions varies, from days to months and sometimes a year. We illustrate this in Figure 1 using a sample medical record segment for one patient. Notice that the time difference between records varies from one month to a few months. Such varying time gaps could be indicative of certain impending disease conditions. For instance, frequent admissions might indicate a severe health problem and the records of those visits provide a source to study progression of the condition. On the other hand, if there are months between the two successive records, dependency on the previous memory should not play an active role to predict the current outcome.

To address the aforementioned challenges in patient subtyping, we propose an integrated approach to identify patient subtypes using a novel Time-Aware LSTM (T-LSTM), which is a modified LSTM architecture that takes the elapsed time into consideration between the consecutive elements of a sequence to adjust the memory content of the unit. T-LSTM is designed to incorporate the time irregularities in the memory unit to improve the performance of the standard LSTM. The main contributions of this paper are summarized below:

- A novel LSTM architecture (T-LSTM) is proposed to handle time irregularities in sequences. T-LSTM has forget, input, output gates of the standard LSTM, but the memory cell is adjusted in a way that longer the elapsed time, smaller the effect of the previous memory to the current output. For this purpose, elapsed time is transformed into a weight using a time decay function. The proposed T-LSTM learns a neural network that performs a decomposition of the cell memory into short and long-term memories. The short-term memory is discounted by the decaying weight before combining it with the long-term counterpart. This subspace decomposition approach does not change the effect of the current input to the current output, but alters the effect of the previous memory on the current output.
- An unsupervised patient subtyping approach is proposed based on clustering the patient population by utilizing the proposed T-LSTM unit. T-LSTM is used to learn a single representation from the temporal patient data in an auto-encoder setting. The proposed T-LSTM auto-encoder maps sequential records of patients to a powerful representation capturing the dependencies between the elements in the presence of time irregularities. The representations learned by the T-LSTM auto-encoder are used to cluster the patients by using the k-means algorithm.

Supervised and unsupervised experiments on both synthetic and real world datasets show that the proposed T-LSTM architecture performs better than standard LSTM unit to learn discriminative representations from sequences with irregular elapsed times.

The rest of the paper is organized as follows: related literature survey is summarized in Section 2, technical details of the proposed approach are explained in Section 3, experimental results are presented in Section 4, and the study is concluded in Section 5.

2 RELATED WORK

Computational Subtyping with Deep Networks. A similar idea as presented in this study was proposed in [25], but for supervised problem settings. Pham et al. introduced an end-to-end deep network to read EHRs, saves patient history, infers the current state and predicts the future. Their proposed approach, called “Deep-Care”, used LSTM for multiple admissions of a patient, and also addressed the time irregularities between the consecutive admissions. A single vector representation was learned for each admission and was used as the input to the LSTM network. Forget gate of standard LSTM unit was modified to account for the time irregularity of the admissions. In our T-LSTM approach, however the memory cell is adjusted by the elapsed time. The main aim of [25] was answering the question “What happens next?”. Therefore, the authors of [25] were dealing with a supervised problem setting whereas we deal with an unsupervised problem setting.

There are several studies in the literature using RNNs for supervised tasks. For instance, in [14], authors focused on patients suffering from kidney failure. The goal of their approach was to predict whether a patient will die, the transplant will be rejected,
or transplant will be lost. For each visit of a patient, the authors tried to answer the following question: which one of the three conditions will occur both within 6 months and 12 months after the visit? RNN was used to predict these aforementioned endpoints. In [22], LSTM was used to recognize patterns in multivariate time series of clinical measurements. Subtyping clinical time series was posed as a multi-label classification problem. Authors stated that diagnostic labels without timestamps were used, but timestamped diagnoses were obtained. LSTM with a fully connected output layer was used for the multi-label classification problem.

In [10] authors aimed to make predictions in a similar way as doctors do. RNN was used for this purpose and it was fed by the patient’s past visits in a reverse time order. The way RNN was utilized in [10] is different than its general usage. There were two RNNs, one for visit-level and the other for variable-level attention mechanisms. Thus, the method proposed in [10] could predict the diagnosis by first looking at the more recent visits of the patient, and then determining which visit and which event it should pay attention.

Another computational subtyping study [9] learned a vector representation for patient status at each time stamp and predicted the diagnosis and the time duration until the next visit by using this representation. Authors proposed a different approach to incorporate the elapsed time in their work. A softmax layer was used to predict the diagnosis and a ReLU unit was placed at the top of the GRU to predict the time duration until the next visit. Therefore, the elapsed time was not used to modify the GRU network architecture but it was concatenated to the input to be able to predict the next visit time. On the other hand, authors in [4] aimed to learn patient similarities directly from temporal EHR data for personalized predictions of Parkinson’s disease. GRU unit was used to encode the similarities between the sequences of two patients and dynamic time warping was used to measure the similarities between temporal sequences.

A different approach to computational subtyping was introduced in [11]. Their method, called Med2Vec, was proposed to learn a representation for both medical codes and patient visits from large scale EHRs. Their learned representations were interpretable, therefore Med2Vec did not only learn representations to improve the performance of algorithms using EHRs but also to provide interpretability for physicians. While the authors did not use RNN, they used a multi-layer perceptron to generate a visit representation for each visit vector.

**Auto-Encoder Networks.** The purpose of our study is patient subtyping which is an instance of unsupervised learning or clustering, therefore we need to learn powerful representations of the patient sequences that can capture the dependencies and the structures within the sequence. One of the ways to learn representations by deep networks is to use auto-encoders. Encoder network learns a single representation of the input sequence and then the decoder network reconstructs the input sequence from the representation learned by the encoder at the end of the input sequence. In each iteration, reconstruction loss is minimized so that the learned representation is effective to summarize the input sequence. In [26] LSTM auto-encoders were used to learn representations for video sequences. Authors tested the performance of the learned representation on supervised problems and showed that the learned representation is able to increase the classification accuracy.

Auto-encoders are also used to generate a different sequence by using the representation learned in the encoder part. For instance, in [7], one RNN encodes a sequence of symbols into a vector representation, and then the decoder RNN map the single representation into another sequence. Authors of [7] showed that their proposed approach can interpret the input sequence semantically and can learn its meaningful representation syntactically.

### 3 METHODOLOGY

#### 3.1 Time-Aware Long Short Term Memory

**3.1.1 Long Short-Term Memory (LSTM).** Recurrent neural network (RNN) is a deep network architecture where the connections between hidden units form a directed cycle. This feedback loop enables the network to keep the previous information of hidden states as an internal memory. Therefore, RNNs are preferred for problems where the system needs to store and update the context information [3]. Approaches such as Hidden Markov Models (HMM) have also been used for similar purposes, however there are distinctive properties of RNNs that differentiates them from conventional methods such as HMM. For example, RNNs do not make the assumption of Markov property and they can process variable length sequences. Furthermore, in principle, information of past inputs can be kept in the memory without any limitation on the time in the past. However, optimization for long-term dependencies is not always possible in practice because of vanishing and exploding gradient problems where the value of gradient becomes too small and too large, respectively. To be able to incorporate the long-term dependencies without violating the optimization process, variants of RNNs have been proposed. One of the popular variants is Long Short-Term Memory (LSTM) which is capable of handling long-term dependencies with a gated structure [18].

A standard LSTM unit comprises of forget, input, output gates, and a memory cell, but the architecture has the implicit assumption of uniformly distributed elapsed time between the elements of a sequence. Therefore, the time irregularity, which can be present in a longitudinal data, is not integrated into the LSTM architecture. For instance, the distribution of the events in a temporal patient record is highly non-uniform such that the time gap between records can vary from days to years. Given that the time passed between two consecutive hospital visits is one of the sources of decision making in the healthcare domain, an LSTM architecture which takes irregular elapsed times into account is required for temporal data. For this purpose, we propose a novel LSTM architecture, called Time-Aware LSTM (T-LSTM), where the time lapse between successive records is included in the network architecture. Details of T-LSTM are presented in the next section.

**3.1.2 Time-Aware LSTM (T-LSTM).** Regularity of the duration between consecutive elements of a sequence is a property that does not always hold. One reason of the variable elapsed time is the nature of the EHR datasets, where the frequency and the number of patient records are quite unstructured. Another reason is missing information in the longitudinal data. In case of the missing data, elapsed time irregularity impacts predicting the trajectory of the
visits can be weeks, months and years. If there are years between latter content. To achieve this, we propose to use a non-increasing time between successive elements to weight the short-term memory of the current output. Therefore, the dependence on the previous record is huge, it means there is no new information recorded for the patient for a long time. Therefore, the dependence on the previous record should be discounted. A network is proposed to incorporate the elapsed time information into the standard LSTM architecture to be able to capture the temporal dynamics of sequential data with time irregularities. The proposed T-LSTM architecture is given in Figure 2 where the input sequence is represented by the temporal patient data. Elapsed time between two immediate records of a patient can be quite irregular. For instance, time between two consecutive admissions/hospital visits can be weeks, months and years. If there are years between two successive records, then the dependency on the previous record is not significant enough to affect the current output, therefore the contribution of the previous memory to the current state should be discounted. The major component of the T-LSTM architecture is the subspace decomposition applied on the memory of the previous time step. While the amount of information contained in the memory of the previous time step is being adjusted, we do not want to lose the global profile of the patient. In other words, long-term effects should not be discarded entirely, but the short-term memory should be adjusted proportional to the amount of time span between the records at time step $t$ and $t-1$. If the gap between time $t$ and $t-1$ is huge, it means there is no new information recorded for the patient for a long time. Therefore, the dependence on the short-term memory should not play a significant role in the prediction of the current output. T-LSTM applies the memory discount by employing the elapsed time between successive elements to weight the short-term memory content. To achieve this, we propose to use a non-increasing function of the elapsed time which transforms the time lapse into an appropriate weight. Mathematical expressions of the subspace decomposition procedure are provided in Equation Current hidden state. First, short-term memory component ($C_{t-1}^S$) is obtained by a network. Note that this decomposition is data-driven and the parameters of the decomposition network are learned simultaneously with the rest of network parameters by back-propagation. There is no specific requirement for the activation function type of the decomposition network. We tried several functions but did not observe a drastic difference in the prediction performance of the T-LSTM unit, however tanh activation function performed slightly better. After the short-term memory is obtained, it is adjusted by the elapsed time weight to obtain the discounted short-term memory ($C_{t-1}^S$). Finally, to compose the adjusted previous memory (Adjusted previous memory) the complement subspace of the long-term memory ($C_{t-1}^T = C_{t-1} - C_{t-1}^S$) is combined with the discounted short-term memory. Subspace decomposition stage of the T-LSTM is followed by the standard gated architecture of the LSTM. Detailed mathematical expressions of the proposed T-LSTM architecture are given below:

$$
C_{t-1}^S = \tanh (W_d C_{t-1} + b_d) \quad \text{(Short-term memory)}
$$

$$
C_{t-1}^S = C_{t-1}^S \ast g (\lambda_t) \quad \text{(Discounted short-term memory)}
$$

$$
C_{t-1}^T = C_{t-1} - C_{t-1}^S \quad \text{(Long-term memory)}
$$

$$
C_{t-1}^C = C_{t-1}^T + C_{t-1}^S \quad \text{(Adjusted previous memory)}
$$

$$
f_t = \sigma \left( W_f x_t + U_f h_{t-1} + b_f \right) \quad \text{(Forget gate)}
$$

$$
i_t = \sigma \left( W_i x_t + U_i h_{t-1} + b_i \right) \quad \text{(Input gate)}
$$

$$
o_t = \sigma \left( W_o x_t + U_o h_{t-1} + b_o \right) \quad \text{(Output gate)}
$$

$$
\dot{C} = \tanh \left( W_c x_t + U_c h_{t-1} + b_c \right) \quad \text{(Candidate memory)}
$$

The major component of the T-LSTM architecture is given in Figure 2 where the input record and the elapsed time at the current time step. The time lapse between the records at time $t-1$, $t$ and $t+1$ can vary from days to years in healthcare domain. T-LSTM decomposes the previous memory into long and short term components and utilizes the elapsed time ($\Delta t$) to discount the short term effects.
where $x_t$ represents the current input, $h_{t-1}$ and $h_t$ are previous and current hidden states, and $C_{t-1}$ and $C_t$ are previous and current cell memories. $\{W_f, U_f, b_f\}$, $\{W_i, U_i, b_i\}$, and $\{W_o, U_o, b_o\}$ are the network parameters of the forget, input, output gates and the candidate memory, respectively. $\{W_q, b_q\}$ are the network parameters of the subspace decomposition. Dimensionalities of the parameters are determined by the input, output and the chosen hidden state dimensionalities. $\Delta_t$ is the elapsed time between $x_{t-1}$ and $x_t$ and $g(\cdot)$ is a heuristic decaying function such that the larger the value of $\Delta_t$, less the effect of the short-term memory. Different types of monotonically non-increasing functions can be chosen for $g(\cdot)$ according to the measurement type of the time durations for a specific application domain. If we are dealing with time series data such as videos, the elapsed time is generally measured in seconds. On the other hand, if the elapsed time varies from days to years as in the healthcare domain, we need to convert the time lapse of successive elements to one type, such as days. In this case, the elapsed time might have large numerical values when there are years between two consecutive records. As a guideline, $g(\Delta_t) = 1/\Delta_t$ can be chosen for datasets with small amount of elapsed time and $g(\Delta_t) = 1/\log(e + \Delta_t)$ [25] is preferred for datasets with large elapsed times.

In the literature, studies proposing different ways to incorporate the elapsed time into the learning process can be encountered. For instance, elapsed time was used to modify the forget gate in [25]. In T-LSTM, one of the reasons behind adjusting the memory cell instead of the forget gate is to avoid any alteration of the current input’s effect to the current output. The current input runs through the forget gate and the information coming from the input plays a role to decide how much memory we should keep from the previous cell. As can be seen in the expressions of Current memory and Current hidden state in Equation 1, modifying the forget gate directly might eliminate the effect of the input to the current hidden state. Another important point is that, the subspace decomposition enables us to selectively modify the short-term effects without losing the relevant information in the long-term memory. Section 3.4 shows that the performance of T-LSTM is improved by modifying the forget gate, which is named as Modified Forget Gate LSTM (MF-LSTM) in this paper. Two approaches are adopted from [25] for comparison. First approach, denoted by MF1-LSTM, multiplies the output of the forget gate by $g(\Delta_t)$ such as $f_t = g(\Delta_t) \times f_t$, whereas MF2-LSTM utilizes a parametric time weight such as $f_t = \alpha (W_f x_t + U_f h_{t-1} + Q_f q_{\Delta_t} + b_f)$ where $q_{\Delta_t} = \left( \frac{\Delta_t}{2^3}, \left( \frac{\Delta_t}{2^5} \right)^2, \left( \frac{\Delta_t}{2^7} \right)^3 \right)$ when $\Delta_t$ is measured in days similar to [25].

Another idea to handle the time irregularity could be imputing the data by sampling new records between two consecutive time steps to have regular time gaps and then applying LSTM on the augmented data. However, when the elapsed time is measured in days, so many new records have to be sampled for the time steps which have years in between. Secondly, the imputation approach might have a serious impact on the performance. A patient record contains detailed information and it is hard to guarantee that the imputed records reflect the reality. Therefore, a change in the architecture of the regular LSTM to handle time irregularities is suggested.

### 3.2 Patient Subtyping with T-LSTM Auto-Encoder

In this paper, patient subtyping is posed as an unsupervised clustering problem since we do not have any prior information about the groups inside the patient cohort. An efficient representation summarizing the structure of the temporal records of patients is required to be able to cluster temporal and complex EHR data. Auto-encoders provide an unsupervised way to directly learn a mapping from the original data [2]. LSTM auto-encoders have been used to encode sequences such as sentences [33] in the literature. Therefore, we propose to use T-LSTM auto-encoder to learn an effective single representation of the sequential records of a patient. T-LSTM auto-encoder has T-LSTM encoder and T-LSTM decoder units with different parameters which are jointly learned to minimize the reconstruction error. The proposed auto-encoder can capture the long and the short term dependencies by incorporating the elapsed time into the system and learn a single representation which can be used to reconstruct the input sequence. Therefore, the mapping learned by the T-LSTM auto-encoder maintains the temporal dynamics of the original sequence with variable time lapse.

In Figure 3, a single layer T-LSTM auto-encoder mechanism is given for a small sequence with three elements $[X_1, X_2, X_3]$. The hidden state and the cell memory of the T-LSTM encoder at the end of the input sequence are used as the initial hidden state and the memory content of the T-LSTM decoder. First input element and the elapsed time of the decoder are set to zero and its first output is the reconstruction $(\hat{X}_3)$ of the last element of the original sequence $(X_3)$. When the reconstruction error $E_r$ given in Equation 1 is minimized, T-LSTM encoder is applied to the original sequence to obtain the learned representation, which is the hidden state of the encoder at the end of the sequence.

$$E_r = \sum_{i=1}^{L} ||X_i - \hat{X}_i||_2^2, \quad (1)$$

where $L$ is the length of the sequence, $X_i$ is the $i$th element of the input sequence and $\hat{X}_i$ is the $i$th element of the reconstructed sequence. The hidden state at the end of the sequence carries concise information about the input such that the original sequence can be reconstructed from it. In other words, representation learned by the encoder is a summary of the input sequence [8]. The number of layers of the auto-encoder can be increased when the input dimension is high. A single layer auto-encoder requires more number of iterations to minimize the reconstruction error when the learned representation has a lower dimensionality compared to the original input. Furthermore, learning a mapping to low dimensional space requires more complexity in order to capture more details of the high dimensional input sequence. In our experiments, a two layer T-LSTM auto-encoder, where the output of the first layer is the input of the second layer, is used because of the aforementioned reasons.

Given a single representation of each patient, patients are grouped by the $k$-means clustering algorithm. Since we do not make any assumption about the structure of the clusters, the simplest clustering
Algorithm, k-means, is preferred. In Figure 3, a small illustration of clustering the patient cohort for 8 patients is shown. In this figure, learned representations are denoted by $R$. If $R$ has the capability to represent the distinctive structure of patient sequence, then clustering algorithm can group patients with similar features (diagnoses, lab results, medications, conditions, and so on) together. Thus, each patient group has a subtype, which is a collection of common medical features present in the cluster. Given a new patient, learned T-LSTM encoder is used to find the representation of the patient and the subtype of the cluster which gives the minimum distance between the cluster centroid and the new patient’s representation is assigned to the new patient. As a result, T-LSTM auto-encoder learns powerful single representation of temporal patient data that can be easily used to obtain the subtypes in the patient population.

4 EXPERIMENTS

In this section, experimental results on synthetic and real world datasets are reported. For synthetic data, two sets of experiments were conducted such as a classification task on a publicly available synthetic EHR dataset and a clustering task with auto-encoder setting on a randomly generated synthetic data. Comparisons between T-LSTM, MF1-LSTM, MF2-LSTM [25], LSTM, and logistic regression are made. The application of T-LSTM auto-encoder on between T-LSTM, MF1-LSTM, MF2-LSTM [25], LSTM, and logistic regression are made. The application of T-LSTM auto-encoder on between T-LSTM, MF1-LSTM, MF2-LSTM [25], LSTM, and logistic regression are made. The application of T-LSTM auto-encoder on between T-LSTM, MF1-LSTM, MF2-LSTM [25], LSTM, and logistic regression are made. The application of T-LSTM auto-encoder on during experiments. All the weights were learned simultaneously and in data-driven manner. Same network settings and parameters were used for all the deep methods for comparison. Therefore, fixed number of epochs were chosen during the experiments instead of using a stopping criteria. Since there are variable size sequences in longitudinal patient data, batches with same sequence sizes were generated instead of padding the original sequences with zero to make every sequence same length. In this study, we did not use the publicly available large scale ICU dataset, MIMIC [19]. MIMIC is an ICU data, therefore sequence length for the majority of patients is very small such as one or two admissions. Even though MIMIC is an important public source for healthcare research, it is not suitable for our purpose such that very short sequences do not enable us to analyze long and short term dependencies and the effect of the elapsed time irregularities.

Table 1: Supervised synthetic EHR experimental results, average AUC of testing on 10 different splits. Training and testing ratio was chosen as 70% and 30%, respectively.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Avg. Test AUC</th>
<th>Stdev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>T-LSTM</td>
<td><strong>0.91</strong></td>
<td>0.01</td>
</tr>
<tr>
<td>MF1-LSTM</td>
<td>0.87</td>
<td>0.02</td>
</tr>
<tr>
<td>MF2-LSTM</td>
<td>0.82</td>
<td>0.09</td>
</tr>
<tr>
<td>LSTM</td>
<td>0.85</td>
<td>0.02</td>
</tr>
<tr>
<td>LR</td>
<td>0.56</td>
<td>0.01</td>
</tr>
</tbody>
</table>

4.1 Synthetic Dataset

4.1.1 Supervised Experiment. In this section, we report experimental results for a supervised task on an artificially generated EHR data which can be found in 2. The aforementioned data has electronic records of up to 100,000 patients with lab results, diagnoses, and start and end dates of the admissions. Each patient has a unique patient ID similar to real world EHR data. We refer to the reference study [20] for further details of the data generation process. Although the dataset is artificially generated, it contains similar characteristics as a real EHR data. In this experiment, target diagnoses was Diabetes Mellitus and the task was a binary classification problem. Input of the network was the sequence of admissions and the output was the predicted label as one-hot vector. Therefore, regular recurrent network setting was utilized for this task instead of auto-encoder. Feature of one admission was a multi-hot vector containing the diagnoses given in the corresponding admission and the vocabulary size was 529. For this purpose, 6,730 patients were sampled with an average of 4 admissions. For this task, a single layer T-LSTM, MF1-LSTM and MF2-LSTM networks were tested to compare the performance based on area under ROC curve (AUC) metric for 50 epochs. In this experiment, number of hidden and softmax layer neurons were chosen as 1028 and 512, respectively. In addition, performance of the traditional logistic regression (LR) classifier was also analyzed. In logistic regression experiments, admissions were aggregated for each patient without incorporating the elapsed time. We also tried to incorporate the elapsed time as a weight by using the same non-increasing function used in T-LSTM during the aggregation of admissions. However, this approach did not improve the performance in our case. The results are summarized in Table 1.

As it can be seen from the Table 1, T-LSTM has a better performance than the baseline approaches. The way to represent the sequential data could be improved further for logistic regression, but aggregation of the admissions for each patient did not perform well for this task. Supervised experiments show that LSTM networks can enable us to leverage the time aspect of the EHR data

---

1 Available at https://github.com/illidanlab/T-LSTM

2 http://www.emrbots.org/
mean RI

<table>
<thead>
<tr>
<th>Method</th>
<th>Mean RI</th>
<th>Std</th>
</tr>
</thead>
<tbody>
<tr>
<td>T-LSTM</td>
<td>0.96</td>
<td>0.03</td>
</tr>
<tr>
<td>MF1-LSTM</td>
<td>0.85</td>
<td>0.13</td>
</tr>
<tr>
<td>LSTM</td>
<td>0.90</td>
<td>0.09</td>
</tr>
</tbody>
</table>

better. In addition, modifying the cell memory yields a better classification performance. According to our observation, MF1-LSTM and MF2-LSTM have better and sometimes similar results as the traditional LSTM for the tasks in our experiments.

4.1.2 Unsupervised Experiment. In this experiment, we investigate the expressive power of the representation learned from the T-LSTM auto-encoder. For this purpose, a synthetic data was randomly generated and the clustering results were evaluated. Since we know the ground truth of the synthetic data, we computed the Rand index (RI), given in Equation 2 [23], of the clustering to observe the discriminative power of the learned representations. A large value of Rand index indicates that the learned representations are clustered close to the ground truth.

$$RI = \frac{(TP + TN)}{(TP + FP + FN + TN)},$$

where $TP$, $TN$, $FP$, $FN$ are true positive, true negative, false positive and false negative, respectively. Note that $0 \leq RI \leq 1$.

The results on a synthetic dataset containing 4 clusters generated from a mixture of normal distributions with four different means and the same covariance are reported. A data point in the synthetic dataset is a sequence of vectors and the values of the sequences are increasing with time. Some of the elements in the sequences are discarded randomly to introduce unstructured elapsed time. Therefore, T-LSTM is a suitable approach for prediction and clustering of PPMI dataset.

In our experiments, we used the pre-processed PPMI data of 654 patients given in [4]. Che et al. [4] collected patients with Idiopathic PD or non PD, imputed missing values, used one-hot feature form for categorical values, and encoded data abnormalities as 1 and 0. As a result, dataset we used has 15,636 records of 654 patients with an average of 25 sequences (minimum sequence length is 3). Authors of [4] also categorized data as features and targets, where the features are related to patient characteristics and the targets correspond to the progression of PD. A total of 319 features consist of motor symptoms/complications, cognitive functioning, autonomic symptoms, psychotic symptoms, sleep problems, depressive symptoms, and hospital anxiety and depression scale. A total of 82 targets are related to motor sign, motor symptom, cognition, and other non-motor factors [4]. Summary of the PPMI data used in this paper can be found in Table 3.

As it can be seen in Table 3, the elapsed time is measured as months. From 1 month to nearly 2 years gap between successive records of patients is encountered in the dataset. Several experiments were conducted on PPMI data to show the performance of the proposed subtyping approach.

4.2 Parkinson’s Progression Markers Initiative (PPMI) Data

In this section, we present experimental results for a real world dataset. Parkinson’s Progression Markers Initiative (PPMI) is an observational clinical and longitudinal study comprising of evaluation of people with Parkinson’s disease (PD), those people with high risk, and those who are healthy [12]. PPMI aims to identify biomarkers of the progression of Parkinson’s disease. PPMI data is a publicly available dataset which contains clinical and behavioral assessments, imaging data, and biospecimens, therefore PPMI is a unique archive of PD [12]. As with many EHRs, PPMI is a longitudinal dataset with unstructured elapsed time. Therefore, T-LSTM is a suitable approach for prediction and clustering of PPMI dataset.

In Figure 4 different colors denote ground truth assignments of different clusters. Representations learned by T-LSTM yields more compact groups in the 2-D space leading to a more accurate clustering result compared to the standard LSTM and MF1-LSTM. As it can be observed from the Figures 4c and 4b, directly multiplying the forget gate with the time coefficient does not always enables a modification which leverages the time irregularity in our experiments. Even though MF1-LSTM produced a higher Rand index, there are examples, such as Figure 4, where LSTM actually learns a better representation than MF1-LSTM. The change in the objective values of T-LSTM, MF1-LSTM and LSTM with respect to the number of epochs are also compared in Figure 5 for the trial illustrated in Figure 4. It is observed that the modifications related to the time irregularity does not affect the convergence of the original LSTM network in a negative way.
Figure 4: Illustration of the clustering results. Different colors denote ground truth assignments of different clusters. T-LSTM auto-encoder learns a mapping for the sequences such that 4 separate groups of points can be represented in the 2-D space.

Figure 5: Change in the objective values of T-LSTM, MF1-LSTM and LSTM with respect to 500 epochs. It is observed that the modifications related to the time irregularity does not deteriorate the convergence of the original LSTM network.

Table 3: Details of PPMI data used in this study. Elapsed time encountered in the data is measured in months and it varies between 1 month to nearly 2 years. Here, the elapsed time interval is not the time interval of PPMI data recording, but elapsed times seen in records of individual patients.

<table>
<thead>
<tr>
<th>Number of Patients</th>
<th>Elapsed Time Interval</th>
<th>Average Sequence Length</th>
<th>Feature Dimensionality</th>
<th>Target Dimensionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>654</td>
<td>[1, 26]</td>
<td>25</td>
<td>319</td>
<td>82</td>
</tr>
</tbody>
</table>

Table 4: Average mean square error (MSE) for 10 different train-test splits for T-LSTM, LSTM, MF1-LSTM, and MF2-LSTM. T-LSTM yielded a better result than the standard LSTM in the presence of the unstructured time gaps. Elapsed time was multiplied by 30 while applying MF2-LSTM since the time lapse is measured in months.

<table>
<thead>
<tr>
<th>MSE</th>
<th>T-LSTM</th>
<th>MF1-LSTM</th>
<th>MF2-LSTM</th>
<th>LSTM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>0.50</td>
<td>0.53</td>
<td>0.51</td>
<td>0.51</td>
</tr>
<tr>
<td>Std</td>
<td>0.018</td>
<td>0.017</td>
<td>0.012</td>
<td>0.017</td>
</tr>
</tbody>
</table>

Table 5: The main observation about the target features in Table 5 is that they are related to the effects of Parkinson’s disease on the muscle control such as finger tapping, rigidity, and hand movements. In addition, T-LSTM predicted the target value of Bradykinesia, which encompasses several of the problems related to movement, and MoCA (Montreal Cognitive Assessment) Total Score, which assesses different types of cognitive abilities with lower error than other methods. This result shows that the reported target features are sensitive to elapsed time irregularities and discounting the short-term effects by the subspace decomposition of memory cell helps to alleviate this sensitivity.

4.2.2 Patient Subtyping of PPMI Data. In this experiment, T-LSTM auto-encoder was used to obtain subtypes of the patients in the PPMI dataset. The T-LSTM encoder was used to learn a representation from the input feature sequence of each patient and the T-LSTM decoder generated the target sequence. Parameters of the auto-encoder were learned to minimize the squared error between the original target sequence and the predicted target sequence. The learned representations were used to cluster the patients by the k-means algorithm as discussed before.

Since we do not know the ground truth for the clustering, we conducted a statistical analysis to assess the subtyping performance. For this purpose, clustering results were statistically analyzed at the time of 6 years follow-up in the PPMI study. Features including demographics, motor severity measures such as Unified Parkinson’s Disease Rating Scale (MDSUPDRS), Hoehn and Yahr staging (H&Y),
Table 5: Some common target features from PPMI dataset on which T-LSTM performed better than LSTM and MF1-LSTM during 10 trials. These target features are mainly related to the effects of Parkinson’s disease on muscle control.

<table>
<thead>
<tr>
<th>Code</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP3BRADY</td>
<td>Global spontaneity of movement</td>
</tr>
<tr>
<td>NP3RIGR</td>
<td>Rigidity - RUE(Right Upper Extremity)</td>
</tr>
<tr>
<td>NP3FTAPR</td>
<td>Finger Tapping Right Hand</td>
</tr>
<tr>
<td>NP3TTAPR</td>
<td>Toe tapping - Right foot</td>
</tr>
<tr>
<td>NP3PRSPR</td>
<td>Pronation-Supination - Right Hand</td>
</tr>
<tr>
<td>NP3HMOVVR</td>
<td>Hand movements - Right Hand</td>
</tr>
<tr>
<td>NP3RIGN</td>
<td>Rigidity - Neck</td>
</tr>
<tr>
<td>NP2DRES</td>
<td>Dressing</td>
</tr>
<tr>
<td>P3RIGRL</td>
<td>Rigidity - RLE (Right Lower Extremity)</td>
</tr>
<tr>
<td>DFBRADYP</td>
<td>Bradykinesia present and typical for PD</td>
</tr>
<tr>
<td>NP3RTARU</td>
<td>Rest tremor amplitude - RUE</td>
</tr>
<tr>
<td>NP3PTTMAR</td>
<td>Postural tremor - Right Hand</td>
</tr>
<tr>
<td>MCATOT</td>
<td>MoCA Total Score</td>
</tr>
</tbody>
</table>

non-motor manifestations such as depression, anxiety, cognitive status, sleep disorders, imaging assessment such as DaTScan, as well as cerebrospinal fluid (CSF) biomarkers were taken into account. In order to interpret the clustering results in terms of subtyping, we compared the clusters using Chi-square test for the categorical features, F-test for the normal continuous features, Kruskal-Wallis test for the non-normal continuous features, and Fisher’s exact test for the high sparsity features. According to the previous Parkinson’s disease studies, if the p-values of the aforementioned features are less than 0.05, a significant group effect is considered for the associated features [15]. Thus, if a method can obtain higher number of features with small p-values, it means that method provides a more sensible patient subtyping result.

Since we do not know the ground truth groups of the patient population, we tried several k values for the k-means algorithm. We often observed that there were two main clusters, therefore we reported the clustering results for k = 2. We conducted several tests with different parameters. According to our observation, LSTM yielded very few features with p-values less than 0.05 and most of the patients were generally grouped into one cluster. In Table 6, features of small p-values and cluster means of the features are presented for T-LSTM, MF1-LSTM and MF2-LSTM. As it can be seen from the table, T-LSTM has more discriminative features than MF1-LSTM and MF2-LSTM.

In Table 6, high cluster mean indicates that the symptoms of the corresponding feature are more severe for that cluster and the PD patients have lower cluster mean for DaTScan feature. Note that one of the observed features of T-LSTM in Table 6 is MoCA which was predicted better by T-LSTM in the target sequence prediction experiment. Finally, we illustrate the patient subtyping results of T-LSTM with heat map illustration in Figure 6. In this figure, shade of red color represents the cluster mean which is higher than the total mean of the patients and the shades of blue show lower mean values for the corresponding feature with p-value < 0.05. Features Note that the dataset contains healthy subjects as well. It is known that PD patients have lower DaTScan SBR values than healthy subjects [29]. Hence, we can conclude from Figure 6 that subtype II can be considered as PD patients. We can also observe from Figure 6 that cluster means of BJLO and MoCA are very low (darker shades of blue) for subtype I compared to subtype II.

### Table 6: Results of the statistical analysis for T-LSTM, MF1-LSTM and MF2-LSTM. DaTScan1 corresponds to DaTScan SBR-CAUDATE RIGHT, DaTScan2 is DaTScan SBR-CAUDATE LEFT, and DaTScan4 is DaTScan SBR-PUTAMEN LEFT.

<table>
<thead>
<tr>
<th>Feature</th>
<th>P-Value</th>
<th>Cluster1 Mean</th>
<th>Cluster2 Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>BJLO</td>
<td>9.51 x 10^{-8}</td>
<td>16.5</td>
<td>24.7</td>
</tr>
<tr>
<td>MoCA</td>
<td>0.001</td>
<td>40.0</td>
<td>41.2</td>
</tr>
<tr>
<td>DaTScan1</td>
<td>0.042</td>
<td>2.29</td>
<td>2.07</td>
</tr>
<tr>
<td>DaTScan2</td>
<td>0.027</td>
<td>2.31</td>
<td>2.08</td>
</tr>
<tr>
<td>DaTScan4</td>
<td>0.001</td>
<td>1.4</td>
<td>1.1</td>
</tr>
<tr>
<td>CSF-Total tau</td>
<td>0.007</td>
<td>87.9</td>
<td>46.72</td>
</tr>
<tr>
<td>MoCA</td>
<td>2.16 x 10^{-17}</td>
<td>47.5</td>
<td>41.05</td>
</tr>
<tr>
<td>SDM</td>
<td>0.005</td>
<td>58.5</td>
<td>41.5</td>
</tr>
<tr>
<td>HVLT-Retention</td>
<td>0.03</td>
<td>0.84</td>
<td>0.83</td>
</tr>
<tr>
<td>SDM</td>
<td>0.007</td>
<td>36.61</td>
<td>41.68</td>
</tr>
</tbody>
</table>

Figure 6: Heat map illustration of the patient subtyping results of T-LSTM for two clusters. Shade of red represents the cluster mean which is higher than the total mean of the patients and the shades of blue show lower mean values for the corresponding feature with p-value < 0.05.

5 CONCLUSION

In this paper, we propose a novel LSTM unit, called time-aware LSTM (T-LSTM) which can deal with irregular elapsed times between the successive elements of sequential data. Examples include medical records which are complex temporal data with varying sequence lengths and elapsed times, and video sequence with missing frames. T-LSTM does not have any assumption about the elapsed time measure such that the time gap does not have to be measured in days or years and thus it can be adopted by other domains dealing with different types of sequences. T-LSTM adjusts the previous memory content of an LSTM unit by a decaying function of the elapsed time in a way that longer the time lapse, less the influence of the previous memory content on the current output. The proposed T-LSTM was tested for supervised and unsupervised tasks.
on synthetic and real world datasets. Patient subtyping, which can be defined as clustering sequential patient records, was analyzed on a publicly available real world dataset called Parkinson’s Progression Markers Initiative (PPMI). For the subtyping purpose, T-LSTM auto-encoder was used to learn powerful representations for the temporal patient data, and the learned representations were used to cluster the patient population. In future work, we plan to apply the proposed approach to several other real datasets to observe the behaviour of our method for patient populations with different characteristics.

ACKNOWLEDGMENTS

Data used in the preparation of this article were obtained from the Parkinson’s Progression Markers Initiative (PPMI) database (http://www.ppmi-info.org/data). For up-to-date information on the study, visit http://www.ppmi-info.org. PPMI a public-private partnership is funded by the Michael J. Fox Foundation for Parkinson’s Research and funding partners, including Abbvie, Avid, Biogen, Bristol-Mayers Squibb, Covance, GE, Genentech, GlaxoSmithKline, Lilly, Lundbeck, Merck, Meso Scale Discovery, Pfizer, Piramal, Roche, Sanofi, Servier, TEVA, UCB and Golub Capital. This research is supported in part by the Office of Naval Research (ONR) under grants numbers N00014-17-1-2265 (to JZ and AKJ), N00014-14-1-0631 (to JZ and AKJ) and National Science Foundation under grants IIS-1565596 (to JZ), IIS-165597 (to JZ) and IIS-1650723 (to FW).

REFERENCES

A Dynamic Cloud Computing Platform for eHealth Systems

Mehdi Bahrami 1 and Mukesh Singhal 2
Cloud Lab
University of California Merced, USA
Email: 1 IEEE Senior Member, MBahrami@UCMerced.edu; 2 IEEE Fellow, MSinghal@UCMerced.edu

Abstract—Cloud Computing technology offers new opportunities for outsourcing data, and outsourcing computation to individuals, start-up businesses, and corporations in health care. Although cloud computing paradigm provides interesting, and cost effective opportunities to the users, it is not mature, and using the cloud introduces new obstacles to users. For instance, vendor lock-in issue that causes a healthcare system rely on a cloud vendor infrastructure, and it does not allow the system to easily transit from one vendor to another. Cloud data privacy is another issue and data privacy could be violated due to outsourcing data to a cloud computing system, in particular for a healthcare system that archives and processes sensitive data. In this paper, we present a novel cloud computing platform based on a Service-Oriented cloud architecture. The proposed platform can be ran on the top of heterogeneous cloud computing systems that provides standard, dynamic and customizable services for eHealth systems. The proposed platform allows heterogeneous clouds provide a uniform service interface for eHealth systems that enable users to freely transfer their data and application from one vendor to another with minimal modifications. We implement the proposed platform for an eHealth system that maintains patients’ data privacy in the cloud. We consider a data accessibility scenario with implementing two methods, AES and a light-weight data privacy method to protect patients’ data privacy on the proposed platform. We assess the performance and the scalability of the implemented platform for a massive electronic medical record. The experimental results show that the proposed platform have not introduce additional overheads when run data privacy protection methods on the proposed platform.

Keywords—Cloud Computing; Data Security; Data Privacy; eHealth Platform; Dynamic Cloud Computing Architecture;

I. INTRODUCTION

Cloud computing offers a new technology to multidiscipline fields to establish a virtual IT department via the Internet [1, 2]. The cloud computing offers different virtual services like traditional IT department, such as storage, stream server and database server. The cloud provides a cost effective model through pay-per-use that allows each individual or businesses in healthcare start a cloud based service with minimum investment [1, 2]. However, the cloud computing has several major issues [1, 3, 4, 5] for an eHealth system which are discussed as follows.

Migration Issue: Data and application migration is one of the major issues when users decide to transfer their data and applications from an IT department to a cloud computing system or from one cloud computing to another. Migration may causes several sub-issues, such as data security issue. For instance, a user who used a regular application based on a specific Application Programming Interface (API) could have some issue when the application transfer to a cloud computing system that needs to redefine or modify the security functions of the API in order to use the cloud. Each cloud computing system offer own services to

Security Issue: Data security refers to accessibility of stored data to only authorized users, and network security refers to accessibility of transfer of data between two authorized users through a network. Since cloud computing uses the Internet as part of its infrastructure, stored data on a cloud is vulnerable to both a breach in data and network security.

Data Privacy: Users have to outsource their data to an untrusted cloud vendor (e.g., public cloud vendors) in order to use the cloud computing benefits. In addition of data and network hack issues in cloud computing, data privacy could be violated by other users, malicious applications or even the cloud vendor when users share their data with a cloud vendor. Data privacy becomes one of the major challenges in outsourcing data to the cloud. Data encryption methods allow users to avoid exposing the original data to the cloud vendors. However, encryption for each single original data is not cost effective or feasible for some machines, such as mobile devices. For example, some mobile devices in eHealth systems have limited resources, such as CPU, RAM and battery power.

II. BACKGROUND

In our previous study, we developed a dynamic cloud computing architecture based on Service-Oriented Architecture (DCCSOA) [4]. The architecture provides a new layer, Template-as-a-Service (TaaS), on the top of a cloud computing system that allows a cloud vendor to standardize its cloud services by defining TaaS services. TaaS is divided into two sub-layers: front-end (FTaaS) that allows different cloud vendors to define a generic and standard cloud service, and back-end (BTaaS) that allows a cloud vendor to bind a defined generic cloud service, FTaaS, to its cloud computing system. In other words, DCCSOA enables different cloud vendors to standardize their services through a uniform interface at FTaaS that allows users to transfer their data and applications from one vendor to another.

In this paper, we use DCCSOA to provide a template, TaaS, for eHealth system. A template allows an eHealth system to use heterogeneous cloud computing systems. It provides flexibility, customizability and standardization for eHealth services that needs to be run on the cloud computing.

As previously discussed, the data security and data privacy are two major issues in cloud computing system for eHealth
systems. We will use a light-weight data privacy method (DPM) [6] that allows clients to scramble the original data on the client side before submitting to the cloud, and AES encryption method on the proposed platform. We evaluate the performance of implemented platform while clients use the methods.

Our contribution in this paper are as follows:

- Propose a platform for eHealth system based on DCCSOA.
- Introduce an eHealth template for the proposed platform that provides a uniform interface for eHealth systems to interact with heterogeneous cloud computing systems.
- Conduct an experiment through DPM and AES on the proposed platform to evaluate the performance and scalability of the proposed platform.

The rest of the paper is organized as follows: In the next section, we introduce the proposed platform based on DCCSOA. We discuss the implementation of the proposed platform in Section IV. We evaluate the behavior of DPM and AES on the proposed platform for a massive healthcare dataset in Section V. We review related work in Section VI, and finally, we conclude our study in Section VII.

III. THE PROPOSED PLATFORM

We consider DCCSOA as the main architecture for the proposed cloud platform. We define an eHealth Template, \( T_{eH} \), for eHealth systems which is divided into the front-end, \( FTaaS_{eH} \), and the back-end, \( BTaaS_{eH} \).

\( FTaaS_{eH} \) provides a generic and a uniform interface with standard services. \( BTaaS_{eH} \) binds specific cloud value-added services to the uniform service interfaces at \( FTaaS_{eH} \).

Figure 1 illustrates a general view of cloud stacks for the proposed platform. A client (end-user) accesses a generic and a uniform cloud service interfaces through an eHealth Client Application. The proposed platform can be simply transferred from a vendor \( V_1 \) to another \( V_2 \) by using the same \( FTaaS_{eH} \) in another cloud but with different \( BTaaS_{eH} \).

\( FTaaS_{eH} \) is a dynamic layer, and it allows cloud vendors to customize their cloud services as a template. First, cloud vendors bind defined generic and uniform services at \( FTaaS_{eH} \) to their value-added services through \( BTaaS_{eH} \). As shown in Equation 1 each service at \( FTaaS_{eH} \) must pass a satisfaction function \( S \) to propose a uniform service interface.

\[
\exists s \in FTaaS_{eH} | Sat(s)
\]

Equation 1 where \( s \) is a service at \( FTaaS_{eH} \) and \( Sat \) is a satisfaction function which is defined as follows:

\[
Sat(s): \mathcal{R} \rightarrow \mathcal{O}
\]

where \( \mathcal{R} \) is a finite set of requirements of \( r \), and \( \mathcal{O} \) is a finite set of corresponding output for each requirement in \( \mathcal{R} \).

The uniform service interface, \( UI \), can be defined as follows:

\[
UI(s) \rightarrow Sat(s_1) \wedge Sat(s_2) \wedge Sat(s_3)
\]

Code I shows an example of how a client accesses \( FTaaS \) through a uniform data access layer with an abstraction on a cloud service (database access in this case). In this code, a client loads a service, \( FTaaS_Service\_Ref \), for accessing services on the proposed platform. Then, the client requests a data access by calling \( GetDataList \) procedure from the web service, and finally, it retrieves the result on an object, \( DataGridView \).

![Figure 1. A view of eHealth template with implementation of DPM and its connection to cloud value-added services](image)

IV. EXPERIMENTAL SETUP

We implemented the proposed platform through a case study based on a defined template for an eHealth system. The proposed platform provides a generic data access at \( FTaaS \) to end-users for accessing to an Electronic Medical Record (EMR). We implemented two methods on the proposed platform to protect patients’ data privacy - one is a light-weight data privacy method (DPM) which is described in [6] and [7] and another method is AES encryption [8]. These methods allows us to assess the performance of the proposed platform.

We consider the following scenario for the implementation of the proposed platform.

“A client requests a data access to an Electronic Medical Record (EMR) which is implemented as a web service at \( FTaaS \). \( FTaaS \) provides a generic, and a uniform function to the client. The request will be submitted from \( FTaaS \) to the \( BTaaS \). Each
retrieved response is processed through two user-data protection methods, DPM and AES encryption. BTaaS is implemented by Windows Communication Foundation (WCF) [9], and it is bounded to a SQL database. We ran different queries at this level, and uses data protection methods to evaluate the performance of the proposed platform. BTaaS’ responses sent to the client at FTaaS by a web service.”

We implemented the proposed platform that includes an eHealth template. The template at the FTaaS enables end-users to interact with data access layer without considering the source of data. In the proposed platform is FTaaS and BTaaS are implemented as a web service, a Windows Communication Foundation (WCF) service, respectively. The services can be easily customized at BTaaS to adapt with heterogeneous cloud computing systems or traditional IT systems.

We used an Artificial Large Medical Dataset as our EMR database that contains records of 100,000 patients, 361,760 admissions; 107,535,387 lab observations, and with the size of 12,494,912 KB (~12.2 GB). We ran 31 different queries on the largest table, lab observations. Each query retrieved different numbers of fields with different size. We ran DPM and AES Encryption at BTaaS to protect patients’ data privacy on each retrieved field. It allows us to assess the performance of the methods on the proposed platform by monitoring the computation time of the methods for each retrieved field from database.

The processed queries in this experiment are based on Select Distinct Top in TSQL language that retrieves data from 6 fields to 30,000 fields with the total queries’ result size from 180 Byte to 911 Mbyte.

In this paper, we are interested in evaluation of both quantity parameters and quality parameters in the proposed platform.

The quality parameters includes the following parameters:

Performance: We consider different workloads to evaluate the performance of a given method on the proposed platform and its performance when the size of workload is increased.

Scalability: A scalable service allows the service to provide the same performance when the number of transactions is increased.

The quality parameters includes the following parameters:

Customization: The higher level of this parameter allows a cloud vendor to customize provided services with minimum modifications.

Independence of services: The higher level of this parameter allows the administrator to freely transfer an eHealth system to another cloud vendor or bring it back to a traditional IT department with minimal service modifications.

Standardization of service: The higher level of this parameter allows an eHealth system to interact with heterogeneous cloud services with minimal modifications.

V. EXPERIMENTAL RESULTS

Figure 2 illustrates the experimental results for the evaluation of the quantity parameters on the proposed platform for an eHealth system. We ran 31 different queries on the EMR database. Each submitted query from FTaaS is processed on the proposed platform to retrieve data from database at BTaaS. The platform is retrieved the response of each query and ran DPM and AES encryption on each retrieved field (result) from BTaaS. Figure 2.a shows the performance of the implemented methods on the proposed platform.

We expect that DPM provide a better performance over AES as described in [8] as well as on the proposed platform. Figure 2.a compares the performance of DPM and AES encryption on the proposed platform. This figure shows that DPM provides a better performance over AES encryption for all query results as we expected.

![Figure 2. Experimental Results](image-url)

1 http://www.emrbots.org retrieved on July 12, 2015
Figure 2.b illustrates the performance of DPM and AES encryption for different size of an input string while the methods are not performed on the proposed platform. We considered each input string as a Unicode character with a size of 16 bits each. X-axis represents the size of input string, and Y-axis represents its response time (milliseconds). In our experiment, we assumed that DPM does not need to generate a set of PRP by accessing to predefined arrays that described in [6].

Figures 2.a and 2.b show that the performance of processing of DPM and AES on the proposed platform (Figure 2.a) is not different from a single string (in Figure 2.b).

Another parameter which can be evaluated is quality parameters that includes service independency and a service standardization.

As described in Code I, a client can access the platform by using the provided generic service. Since the service is independent of the cloud value-added services at the BTaaS, it allows users to interact with the cloud services without concerning about its requirements or type of output of a service. For instance, an application at client side in Scenario 1 retrieves data without understanding the type of database, and the location of the database. The service at FTaaS can be bind to any kind of services at BTaaS.

Different cloud vendors are able to define the similar services at FTaaS in Scenario I that allows an eHealth system use different cloud standardized services.

VI. RELATED WORK

Several cloud-based services and platforms have been developed for eHealth systems. For instance, Fan et al. [10] developed a platform which is used from capturing health care data for processing on the cloud computing. The platform relies on its architecture, and the authors did not describe how the proposed platform can be implemented for different architectures or how it can customize services for heterogeneous clouds. As discussed previously, a dynamic and a customizable cloud platform allows administrators to implement, and to transfer an eHealth system to different cloud computing systems. There is also a vendor lock-in issue [5], if a platform’s services rely on a specific cloud architecture. In another study, Lounis et al. [11] developed a secure cloud architecture which is only focused on wireless sensor networks, and the study has limited work on the architecture. The study does not discussed the architecture features, such as service modifications or dynamic services. Magableh et al. [12] proposed a dynamic rule-based approach without considering the cloud environment. Finally, Hoang et al. [13] focus on mobile users features in their proposed architecture, and the study does not discuss the overall of the architecture. In our study, we proposed a dynamic platform for eHealth system, and we showed how the proposed platform implements a dynamic service at FTaaS.

VII. CONCLUSION

In this paper, we proposed a dynamic cloud platform for an eHealth system based on a cloud SOA architecture, DCSSOA. The proposed platform can be run on the top of heterogeneous cloud computing systems that allows a cloud vendor to customize and standardize services with minimal modifications.

The platform uses a template layer which is divided into FTaaS that allows cloud vendors to define a standard, generic, and uniform service, and BTaaS that allows defined services at BTaaS to bind to the cloud vendor value-added services. In addition, we implemented a data access scenario on the proposed platform with two different methods to evaluate its performance. The first method is a light-weight data privacy method (DPM), and the second is AES encryption method. The evaluation shows that the platform is scalable and the methods which are ran on the platform have not introduce additional overheads.

REFERENCES


Citation (To appear):

Data and text mining
comoRbdity: an R package for the systematic analysis of disease comorbidities

Alba Gutierrez-Sacristan1,2, Álex Bravo1,3, Alexia Giannoula1, Miguel A. Mayer1, Ferran Sanz1 and Laura I. Furlong1,*

1Research Programme on Biomedical Informatics (GRIB), Department of Experimental and Health Sciences (DCEXS), Hospital del Mar Medical Research Institute (IMIM), Universitat Pompeu Fabra (UPF), Barcelona 08003, Spain, 2Department of Biomedical Informatics, Harvard Medical School, Boston, MA 02115, USA and 3Large-Scale Text Understanding Systems Lab, TALN Research Group, Department of Information and Communication Technologies (DTIC), Universitat Pompeu Fabra, Barcelona 08018, Spain

*To whom correspondence should be addressed.

Associate Editor: Janet Kelso

Received on September 1, 2017; revised on March 23, 2018; editorial decision on April 18, 2018; accepted on April 19, 2018

Abstract

Motivation: The study of comorbidities is a major priority due to their impact on life expectancy, quality of life and healthcare cost. The availability of electronic health records (EHRs) for data mining offers the opportunity to discover disease associations and comorbidity patterns from the clinical history of patients gathered during routine medical care. This opens the need for analytical tools for detection of disease comorbidities, including the investigation of their underlying genetic basis.

Results: We present comoRbdity, an R package aimed at providing a systematic and comprehensive analysis of disease comorbidities from both the clinical and molecular perspectives. comoRbdity leverages from (i) user provided clinical data from EHR databases (the clinical comorbidity analysis) and (ii) genotype-phenotype information of the diseases under study (the molecular comorbidity analysis) for a comprehensive analysis of disease comorbidities. The clinical comorbidity analysis enables identifying significant disease comorbidities from clinical data, including sex and age stratification and temporal directionality analyses, while the molecular comorbidity analysis supports the generation of hypothesis on the underlying mechanisms of the disease comorbidities by exploring shared genes among disorders. The open-source comoRbdity package is a software tool aimed at expediting the integrative analysis of disease comorbidities by incorporating several analytical and visualization functions.

Availability and implementation: https://bitbucket.org/ibi_group/comorbidity
Contact: laura.furlong@upf.edu
Supplementary information: Supplementary data are available at Bioinformatics online.

1 Introduction

The co-existence of two or more diseases in the same patient, also known as comorbidity (van den Akker et al., 1996; Valderas et al., 2009) is a matter of public health concern as it has important consequences both for patients and the healthcare system (Gijzen et al., 2001; Valderas et al., 2009). According to several studies, the prevalence of comorbidity varies between ~20% and ~90% (Bonavita and De Simone, 2008; Fortin et al., 2005; Mezzich and Salloum, 2008; Marengoni et al., 2011). This variation is due to the population under study, as well as other characteristics of the study design, such as the definition of comorbidity (van den Akker et al., 1996; Valderas et al., 2009). Although the prevalence of comorbidity increases with age, it is not limited to the elderly population (Doshi-Velez et al., 2014; Jakovljević and Ostojić, 2013; Marengoni et al., 2011;
Taylor et al., 2010). The availability of electronic health records (EHR) for data mining offers the opportunity to discover disease associations and comorbidity patterns from the clinical history of patients gathered during routine medical care (Bagley et al., 2016; Backenroth et al., 2016; Holmes et al., 2011).

In recent years, there has been a growing interest in the re-use of clinical data for research (Jensen et al., 2012). In this context, the availability of tools that enable the analysis of clinical data in a reproducible manner and in a secure environment is key. The development of analytical tools to identify comorbidity patterns from clinical data will enable: (i) the estimation of the prevalence of comorbidities in particular populations, (ii) the stratification of patients according to their comorbidities and (iii) the development of decision support systems in the clinical setting.

In this paper, we introduce comoRbidity, an R package aimed at providing a comprehensive analysis of disease comorbidities from both the clinical and molecular perspectives. comoRbidity leverages from clinical data obtained from EHR databases or health registries (the clinical comorbidity analysis), and from genotype-phenotype information of the diseases under study (the molecular comorbidity analysis) from DisGeNET (Piñero et al., 2017), or provided by the user.

### 2 Design and implementation

comoRbidity aims at expediting the analysis of disease comorbidities by providing several analytical functions and different visualization options to analyze clinical data provided by the user. comoRbidity is based on standard CRAN and Bioconductor classes allowing for full flexibility and integration with other R packages. It runs under Linux, Windows and Mac operating systems.

The R CRAN package parallel (R, 2014) is used to speed up the comorbidity estimation by adjusting the cores according to the user requirements. comoRbidity contains 14 R functions (see Supplementary Table S1 for details) used to process clinical and molecular data to perform the disease comorbidity analysis and visualize the results. The package includes a dataset of artificially generated clinical data (http://www.emrbots.org/) to illustrate the functionalities of the package.

The software implements two types of independent analysis, the clinical comorbidity analysis and the molecular comorbidity analysis. An overview of the workflow of data analysis provided by the package is shown in Figure 1. Each analysis includes three sequential steps:

1. **Data Selection**: From the user’s input data, comoRbidity provides an overview of the data, including a demographic analysis.
based on age and sex, the number of genes associated to the diseases under study, or the number of diseases sharing genes.

ii. Data Analysis: The comorbidity analysis is performed, based on different parameters set by the user.

iii. Results Visualization: The package offers different options for the visualization of the results.

3 Related work

To the best of our knowledge, only few tools have been developed for the analysis of disease comorbidities, namely comoR (Moni et al., 2014), CytoCom (Moni et al., 2015) and medicalRisk (McCormick, 2016). In the R environment, the comoR package (Moni et al., 2014) computes statistically significant associations among diseases based on the US Medicare claims database (Hidalgo et al., 2009) along with several molecular and phenotypic association metrics. The same authors developed CytoCom2 (Moni et al., 2015), a Cytoscape App to visualize and query their disease comorbidity networks (Hidalgo et al., 2009). The medicalRisk R package (McCormick, 2016), can be used to obtain medical risk status from large datasets with diseases encoded in ICD-9-CM, based on mortality predictors such as the Charlson Comorbidity Index and the Elixhauser comorbidity map. Compared to these tools, the main advantage of comoRbidity is the possibility to analyze the user’s own clinical data in his/her private workstation, avoiding any privacy issues concerning the sharing of patient data. In addition, comoRbidity allows any classification to encode diseases, and provides different statistics and functions for assessing comorbidity between disorders. Finally, it allows exploring the genetic basis of disease comorbidities by the analysis of gene-disease association data from DisGeNET (Piñero et al., 2017), or from gene-disease association data provided by the user.

4 Conclusions

The comoRbidity package is a novel, publicly available tool for the processing of healthcare data to identify comorbidity patterns enabling their analysis in a user-friendly and reproducible manner. More importantly, it permits the user to provide its own clinical data, which can be analyzed locally in a secure environment. comoRbidity supports any classification system used to identify diseases and/or phenotypes. In addition, it permits full flexibility to the user in the definition of comorbidity regarding the temporal window considered, the diseases of interest and the use of primary or secondary diagnoses in the analysis, among other aspects. Several analytical and visualization functions are provided including metrics to assess disease associations and their temporal directionality. In addition, it allows performing a molecular analysis of the comorbidities even if no genomic data of the patient is available, by using publicly available information on gene-disease associations, making possible the formulation of hypothesis regarding the etiology of disease comorbidities.

Conflict of Interest: none declared.

References


Recommender Systems
TIETS43

Fairness in Group Recommendations in the Health Domain

Kostas Stefanidis
kostas.stefanidis@uta.fi

Fall 2017
https://coursepages.uta.fi/tiets43/
Motivation (1/2)

The problem:

• Approximately 72 percent of Internet users seek health information online
• It is very difficult for a patient to accurately judge the relevance of some information to his own case

• Doctors and caregivers on the other hand have not the time to provide detailed information to individuals and to small groups of patients

• A real requirement from doctors in iManageCancer EU project: “A tool to automatically recommend useful, high quality documents to similar groups of patients”
We target at improving the opportunities that patients have to inform themselves via the Web about their disease and possible treatments, and providing to them personalized information

- **Deliver relevant information to patients**
  - How?
    - Based on their current profile as represented in their personal healthcare record (PHR) data.

- **Ensure the quality of the presented information**
  - How?
    - Giving medical experts the chance to control the information that is given.
    - Identify once a high quality corpus of interesting documents and those will be automatically recommended to small group of patients according to their problems.
Goals

• A model for group recommendations
  ▫ Both group recommendations and notions of fairness are mostly unexplored in the health domain

• Collaborative filtering approach

• Explore different similarity measures that take into consideration specific health-related information to identify the correct set of similar users for a user in question

• Use different designs for aggregating the recommendations for the group

• Suggestions highly related and fair to the patients of the group
Outline

• Single User Recommendations
  ▫ Single User Rating Model
  ▫ User Similarities
  ▫ Example

• Group Recommendations
  ▫ Group Rating Model
  ▫ Fairness in Group Recommendations
  ▫ Aggregations Designs

• Dataset

• Evaluation
Single User Recommendations
Recommendation Model

\[ I = \{i_1, i_2, \ldots, i_d\} \] : set of items

\[ U = \{u_1, u_2, \ldots, u_n\} \] : set of users

\( rating(u, i) \in [1, 5] \) : the preference / rating of a user \( u \in U \) for an item \( i \in I \)

\( U(i) \): the subset of users that rated an item \( i \)

\( I(u) \): the subset of items rated by a user \( u \)

*But, typically users rate only a few items (and \(|I|\) is too high!)*

For an unrated item \( i \), estimate its relevance for a user \( u \)

\[ relevance(u, i) \]
Single User Recommendations (1/3)

How to estimate \( \text{relevance}(u,i) \)?

- Collaborative filtering idea: use preferences of similar users to \( u \) to produce relevance scores for the items unrated by \( u \)
  
  - Similarity is estimated in terms of some similarity / distance function
  - \( P_u \): the set of similar users to \( u \), or peers, taking into consideration a distance threshold
Relevance computation based on peers

\[
relevance(u, i) = \frac{\sum_{u' \in (P_u \cap U(i))} S(u, u') r(u', i)}{\sum_{u' \in (P_u \cap U(i))} S(u, u')}
\]

After estimating the relevance scores of all unrated items for a user \( u \), the items \( A_u \) with the top-k relevance scores are suggested to \( u \).
Similarity Between Users

Two ways to find similarities between users

• Using the ratings of users
  ▫ Pearson correlation

• Using the Personal Health Profile (PHR) of users
  ▫ Semantic Similarity function
Rating Similarity Function

Similarity based on user ratings

Assumption: if two users have rated documents in a similar way, then we can say that they are similar, since they share the same interests

\[
RatS(u, u') = \frac{\sum_{i \in X} (r(u, i) - \mu_u)(r(u', i) - \mu_{u'})}{\sqrt{\sum_{i \in X} (r(u, i) - \mu_u)^2} \sqrt{\sum_{i \in X} (r(u', i) - \mu_{u'})^2}}
\]

Pearson correlation: where \( X = I(u) \cap I(u') \) denotes the items that both users have rated, \( r(u, i) \) is the rating \( u \) gave to \( i \), and \( \mu_u \) is the mean of the ratings in \( I(u) \)
Semantic Similarity Function

Similarity based on semantic information

Considers the users health problems: represent health problems utilizing the ICD10 ontology

Two steps approach:

- Find the similarity between pairs of “problems” of two users
- Calculate the overall similarity of the users
**Similarity between two health problems (1/7)**

**Step One:** Find similarity between two health problems

<table>
<thead>
<tr>
<th>Code Id</th>
<th>Description</th>
<th>Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>S27</td>
<td>Injury of other and unspecified intrathoracic organs</td>
<td>1</td>
</tr>
<tr>
<td>S29</td>
<td>Other and unspecified injuries of thorax</td>
<td>1</td>
</tr>
<tr>
<td>S27.3</td>
<td>Other injury of bronchus, unilateral</td>
<td>2</td>
</tr>
<tr>
<td>S27.4</td>
<td>Injury of bronchus</td>
<td>2</td>
</tr>
<tr>
<td>S27.43</td>
<td>Laceration of bronchus</td>
<td>3</td>
</tr>
<tr>
<td>S27.49</td>
<td>Other injury of bronchus</td>
<td>3</td>
</tr>
<tr>
<td>S27.491</td>
<td>Other injury of bronchus, unilateral</td>
<td>4</td>
</tr>
<tr>
<td>S27.492</td>
<td>Other injury of bronchus, bilateral</td>
<td>4</td>
</tr>
</tbody>
</table>

Sibling nodes share different similarity, based on the level they reside.
Similarity between two health problems (2/7)

We utilize the ontology tree and assign weights to nodes.

$$weight(A) = w \times 2^{\max \text{Level} - \text{level}(A)}$$

$w = 0.1$
Similarity between two health problems (3/7)

To find similarity between two nodes:

1. Find their Least Common Ancestor
2. Calculate the distance between each node and the LCA
3. Accumulate and normalize those distances

$w = 0.1$
Similarity between two health problems (4/7)

Find their Least Common Ancestor

We allow a node to be descendant of itself.

Examples:

LCA(S27.4, S27.491) = S27.4

LCA(S27, S29.0) = root
Similarity between two health problems (5/7)

Calculate the distance between each node and the LCA

1. Find the path connecting the two nodes
   • In the path that connect the nodes A and B, we include A, but not include B. Example:
     path(S27.491, S27.4) = \{S27.491, S27.49\}
2. Accumulate the weights of the nodes in the path

Examples:
\[\text{dist}(S27.491, S27.4) = 0.3\]
\[\text{dist}(S27.492, root) = 1.5\]
Similarity between two health problems (6/7)

Accumulate and normalize those distances

The final similarity score of two nodes A and B, with LCA(A,B) = C is:

$$simN(A, B) = 1 - \frac{dist(A, C) + dist(B, C)}{maxPath \times 2}$$

where $maxPath$ is the maximum distance in the tree.

- In the ICD10 ontology tree and $w=0.1$, maxPath = 1.5
Similarity between two health problems (7/7)

Examples:

<table>
<thead>
<tr>
<th>Node A</th>
<th>Node B</th>
<th>LCA(A,B)</th>
<th>simN(A,B)</th>
</tr>
</thead>
<tbody>
<tr>
<td>S27</td>
<td>S29</td>
<td>root</td>
<td>1-(0.8+0.8/3)=0.47</td>
</tr>
<tr>
<td>S27.43</td>
<td>S27.49</td>
<td>S27.4</td>
<td>1-(0.2+0.2/3)=0.87</td>
</tr>
<tr>
<td>S27.492</td>
<td>S27.49</td>
<td>S27.49</td>
<td>1-(0 + 0.1/3) = 0.97</td>
</tr>
<tr>
<td>S27.491</td>
<td>S27.492</td>
<td>S27.49</td>
<td>1-(0.1 + 0.1 /3) = 0.93</td>
</tr>
</tbody>
</table>
Overall Similarity between two users

Users have multiple of health problems in their profile. So the final similarity of two users:

Assume users $x$, $y$.
1. Compare a health problem from $x$’s profile with all the health problem from $y$’s profile.
2. Choose the one with the maximum similarity.
3. Do the same for all problem from $x$’s profile.
4. Accumulate and then average those.

$x$’s health problems

<table>
<thead>
<tr>
<th>S27</th>
<th>S29</th>
</tr>
</thead>
</table>

$y$’s health problems

| S27.49 | S27.491 | S29.0 |

• simN(S27, S27.49) = 1-(0+0.6)/3 = 0.8
• simN(S27, S27.491) = 1-(0+0.7)/3 = 0.77
• simN(S27, S29.01) = 1-(0.8+1,2)/3 = 0.34
• simN(S29, S27.49) = 1-(0.8+1.4)/3 = 0.6
• simN(S29, S27.491) = 1-(0.8+1.5)/3 = 0.23
• simN(S29, S29.0) = 1-(0+0.4)/3 = 0.86

SemS(x,y) = (0.8 + 0.86) / 2 = 0.83
## Single User Recommendation: Example (1/3)

<table>
<thead>
<tr>
<th>#</th>
<th>Name</th>
<th>Problems</th>
<th>Ratings – DocId(score)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>John Smith</td>
<td>S27.492</td>
<td>10(4), 15(5), 20(3), 30(3), 22(4), 23(2), 36(5)</td>
</tr>
<tr>
<td>2</td>
<td>Mary Jane</td>
<td>S27.4, S27.49</td>
<td>10(3), 30(5), 16(3), 19(4), 18(3), 17(4), 35(4)</td>
</tr>
<tr>
<td>3</td>
<td>Thomas Murphy</td>
<td>S27.3</td>
<td>16(5), 20(5), 30(2), 25(3), 22(3), 17(5), 36(5)</td>
</tr>
<tr>
<td>4</td>
<td>Scott Wilson</td>
<td>S29.00</td>
<td>25(5), 45(2), 19(2), 17(5), 31(5), 35(5)</td>
</tr>
</tbody>
</table>

**Rating Similarity Function:**

\[
RatS(u, u') = \frac{\sum_{i \in X} (r(u, i) - \mu_u)(r(u', i) - \mu_{u'})}{\sqrt{\sum_{i \in X} (r(u, i) - \mu_u)^2} \sqrt{\sum_{i \in X} (r(u', i) - \mu_{u'})^2}}
\]

For user John Smith: \( \mu_u = 3.71 \)

For user Mia Brown: \( \mu_u = 4 \)

Common documents: 15, 22, 36

\[RatS(\text{John Smith}, \text{Mia Brown}) = 0.897\]

<table>
<thead>
<tr>
<th>User</th>
<th>RatS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mary Jane</td>
<td>-0.993</td>
</tr>
<tr>
<td>Thomas Murphy</td>
<td>0.389</td>
</tr>
<tr>
<td>Scott Wilson</td>
<td>0</td>
</tr>
<tr>
<td>Mia Brown</td>
<td>0.897</td>
</tr>
</tbody>
</table>
## Single user Recommendation Example (2/3)

<table>
<thead>
<tr>
<th>#</th>
<th>Name</th>
<th>Problems</th>
<th>Ratings – DocId(score)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>John Smith</td>
<td>S27.492</td>
<td>10(4), 15(5), 20(3), 30(3), 22(4), 23(2), 36(5)</td>
</tr>
<tr>
<td>2</td>
<td>Mary Jane</td>
<td>S27.4, S27.49</td>
<td>10(3), 30(5), 16(3), 19(4), 18(3), 17(4), 35(4)</td>
</tr>
<tr>
<td>3</td>
<td>Thomas Murphy</td>
<td>S27.3</td>
<td>16(5), 20(5), 30(2), 25(3), 22(3), 17(5), 36(5)</td>
</tr>
<tr>
<td>4</td>
<td>Scott Wilson</td>
<td>S29.00</td>
<td>25(5), 45(2), 19(2), 17(5), 31(5), 35(5)</td>
</tr>
</tbody>
</table>

### Semantic Similarity Function

<table>
<thead>
<tr>
<th>User</th>
<th>Partial Scores</th>
<th>SemS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mary Jane</td>
<td>S27.492 - S27.4 = 0.9</td>
<td>0.967</td>
</tr>
<tr>
<td></td>
<td>S27.492 - S27.49 = 0.967</td>
<td></td>
</tr>
<tr>
<td>Thomas Murphy</td>
<td>S27.492 - S27.3 = 0.634</td>
<td>0.634</td>
</tr>
<tr>
<td>Scott Wilson</td>
<td>S27.492 - S29.01 = 0.034</td>
<td>0.034</td>
</tr>
<tr>
<td>Mia Brown</td>
<td>S27.492 - S29 = 0.234</td>
<td></td>
</tr>
<tr>
<td></td>
<td>S27.492 - S27.49 = 0.967</td>
<td>0.967</td>
</tr>
</tbody>
</table>
Single user Recommendation-Example (3/3)

\[
relevance(u, i) = \frac{\sum_{u' \in (P_u \cap U(i))} S(u, u') r(u', i)}{\sum_{u' \in (P_u \cap U(i))} S(u, u')}
\]

<table>
<thead>
<tr>
<th>User</th>
<th>Sim</th>
<th>Peers</th>
<th>Recommendation List</th>
</tr>
</thead>
<tbody>
<tr>
<td>John Smith</td>
<td>RatS</td>
<td>Thomas Murphy, Mia Brown</td>
<td>18(5), 16(5), 17(5), 28(3), 31(3), 25(3), 45(2)</td>
</tr>
<tr>
<td></td>
<td>SemS</td>
<td>Mary Jane, Thomas Murphy, Mia Brown</td>
<td>18(5), 17(4.4), 35(4), 19(4), 16(3.8), 18(3), 25(3), 28(3), 31(3), 45(2)</td>
</tr>
<tr>
<td>Mary Jane</td>
<td>RatS</td>
<td>Scott Wilson</td>
<td>25(5), 31(5), 45(2)</td>
</tr>
<tr>
<td></td>
<td>SemS</td>
<td>John Smith, Thomas Murphy, Mia Brown</td>
<td>15(5), 36(5), 22(4.1) 20(3.8), 25(3), 28(3), 31(3), 23(2), 45(2)</td>
</tr>
</tbody>
</table>

**John Smith with SemS**

Document 17 voted by:
- Mary Jane - 4
- Thomas Murphy - 5

SemS(John Smith, Mary Jane) = 0.967
SemS(John Smith, Thomas Murphy) = 0.634

\[
relevance(JohnSmith,17) = \frac{0.967 \times 4 + 0.634 \times 5}{0.967 + 0.634} = 4.4
\]
Group Recommendations
Recently, group recommendations have received considerable attention!
Make recommendations to groups of users instead of single users

How:

• Estimate the relevance scores of the unrated items for each user in the group

• Aggregate these predictions to compute the suggestions for the group

\[ relevance^G(G, i) = Aggr_{u \in G}(relevance(u, i)). \]

Different designs regarding the aggregation method can be used – each one carries different semantics
Our goal is to locate suggestions that include data items highly relevant and fair to the patients of a group
  - We actually provide suggestions to a caregiver responsible for a group of patients

The package selection should be fair to all users in the group!
  - I.e., we do not want to have any user $u$ that is the least satisfied user in the group for all items in the recommendations list, that is, all items are \textbf{not} related to $u$
We consider a fairness measure that evaluates the goodness of recommendations.

Given a user $u$ and a set of recommendations $D$, we define the degree of fairness of $D$ for $u$ as:

$$\text{fairness}(u, D) = \frac{|X|}{|D|},$$

where $X = A_u \cap D$ and $A_u$ are the items with the top-$k$ relevant scores for $u$.

Intuitively, the fact that recommendations contain a highly relevant item to $u$, makes both $u$ and his caregiver tolerant to the existence of other items not highly related to the user, considering that there are other members in the group who may be related to these items.
The fairness of a set of recommendations $D$ for a set of users $G$ is defined as follows.

$$\text{fairness}(G, D) = \frac{\sum_{u \in G} \text{fairness}(u, D)}{|G|}.$$ 

The fairness - aware value of $D$ for $G$ is:

$$\text{value}(G, D) = \text{fairness}(G, D) \cdot \sum_{i \in D} \text{relevance}_G(G, i).$$
We consider four Aggregation Designs that can be divided into two groups:

1. Score Based Methods
   1. Minimum
   2. Average

2. Rank Based Methods
   1. Borda
   2. Fair

\[ relevance_G(G, i) = \operatorname{Aggr}_{u \in G}(relevance(u, i)). \]

\textit{Remember: we have already found the individual recommendation lists for the members of the group}
**Minimum**

\[ relevance_G(G, i) = \min_{u \in G} (relevance(u, i)) \]

Assuming we have two members in the group x, y and their corresponding recommendation lists

<table>
<thead>
<tr>
<th>DocId</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>20</td>
<td>4</td>
</tr>
<tr>
<td>30</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DocId</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>4</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
</tr>
<tr>
<td>30</td>
<td>1</td>
</tr>
</tbody>
</table>

We consider that strong user preferences act as a veto

Using the Minimum Aggregation method the group recommendation list will be:

<table>
<thead>
<tr>
<th>DocId</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>4</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
</tr>
<tr>
<td>30</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ \min(5, 2) = 2 \]
Average

Having the same assumption as before:

x’s list

<table>
<thead>
<tr>
<th>DocId</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>20</td>
<td>4</td>
</tr>
<tr>
<td>30</td>
<td>3</td>
</tr>
</tbody>
</table>

y’s list

<table>
<thead>
<tr>
<th>DocId</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>4</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
</tr>
<tr>
<td>30</td>
<td>1</td>
</tr>
</tbody>
</table>

We focus on satisfying the majority of the group members

Using the Average Aggregation method the group recommendation list will be:

<table>
<thead>
<tr>
<th>DocId</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>4</td>
</tr>
<tr>
<td>10</td>
<td>3.5</td>
</tr>
<tr>
<td>30</td>
<td>2</td>
</tr>
</tbody>
</table>

\[
\text{relevance}_G(G, i) = \frac{\sum_{u \in G \text{ relevance}(u, i)}}{|G|}. \]

\[
\frac{(5+2)}{2} = 3.5
\]
Borda

Having the users x, y:

- **x’s list**
  - DocId: 20, Score: 4, Points: 3
  - DocId: 30, Score: 3, Points: 2
  - DocId: 10, Score: 2, Points: 1

- **y’s list**
  - DocId: 10, Score: 5, Points: 3
  - DocId: 20, Score: 4, Points: 2
  - DocId: 30, Score: 1, Points: 1

Using the Borda Aggregation method the group recommendation list will be:

- DocId: 20, Points: 5
- DocId: 10, Points: 4
- DocId: 30, Points: 3

We consider the number of times a document wins.
Targeting at increasing the fairness of the resulting set of recommendations, we introduce also the Fair method, which consists of two phases.

First:
• we consider pairs of users in the group
• A data item $i$ belongs to the top-$k$ suggestions for a group $G$, if, for a pair of users $u_1, u_2 \in G, i \in A_{u_1} \cap A_{u_2}$ and $i$ is the item with the maximum rank in $A_{u_2}$

Second:
• If the value of $k$ (i.e. the number of items that need to be provided by the group recommendation list) is greater than the items already found
  • We construct the rest of $D$, by serially iterating the $A_u$ of the group members and adding the item with the maximum rank that does not already exists in $D$
## Fair (2/2)

Having the users x, y:

- **x’s list**
  - DocId | Score
  - 20   | 4
  - 10   | 3
  - 30   | 2

- **y’s list**
  - DocId | Score
  - 10   | 5
  - 50   | 3
  - 20   | 1

For x:
The item with highest score in y’s list that also exists in x’s is item 10

For y:
The item with highest score in x’s list that also exists in y’s is item 20

We target to increase the fairness of the resulting set.
Dataset
In order to start the group recommendation process we need three things

- A document corpus
- A users-PHR dataset
- A users-ratings dataset
Dataset Acquisition - PHR dataset

• From EMRBots\(^1\) we acquired:

• A 10,000 chimeric patient profiles
  ▫ These profiles contain the health problems for each patient, that are described using the ICD10 ontology.
Dataset Creation - Document Corpus

- Generate a $numDocs$ number of documents, for each first level category of the ICD10 ontology (i.e. for each node that belongs in the first level of the ontology tree)
- For their corresponding keywords we randomly selected $numKeyWords$ words from the description of the nodes in each subsequent subtree
- Randomly select a $popularDocs$ number of documents that will be the most popular

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Explanation</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>numDocs</td>
<td>The number of documents created for each different category of health problems, based on the ICD10 ontology tree</td>
<td>270</td>
</tr>
<tr>
<td>numKeyWords</td>
<td>The number of randomly selected keywords, attached to each document</td>
<td>10</td>
</tr>
<tr>
<td>popularDocs</td>
<td>The number of documents, that will be most popular in each category, in order to simulate a power law distribution</td>
<td>70</td>
</tr>
</tbody>
</table>
Dataset Creation - Ratings Dataset (1/2)

- Divide the patients into groups.
  - *Sparse*: few number of ratings
  - *Medium*: average number of ratings
  - *Dedicated*: a lot of ratings

- Generate items to rate.
  - *healthRelevant*: documents belonging in the same subtree as a user’s health problems
  - *nonRelevant*: the rest

- Generate ratings.
  - Randomly select a value from *[1,5]* to assign to each rating.
## Dataset Creation - Ratings Dataset (2/2)

<table>
<thead>
<tr>
<th>Partitions</th>
<th>Parameter Name</th>
<th>Explanation</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group Partition</td>
<td>Group <em>sparse</em></td>
<td>The number of ratings given by patients in this group is 20 to 100</td>
<td>50% of all patients</td>
</tr>
<tr>
<td></td>
<td>Group <em>medium</em></td>
<td>The number of ratings given by patients in this group is 100 to 250</td>
<td>30% of all patients</td>
</tr>
<tr>
<td></td>
<td>Group <em>dedicated</em></td>
<td>The number of ratings given by patients in this group is 250 to 500</td>
<td>20% of all patients</td>
</tr>
<tr>
<td>Scores Partition</td>
<td>One</td>
<td>The number of ratings that have 1 as value</td>
<td>20% of all ratings</td>
</tr>
<tr>
<td></td>
<td>Two</td>
<td>The number of ratings that have 2 as value</td>
<td>10% of all ratings</td>
</tr>
<tr>
<td></td>
<td>Three</td>
<td>The number of ratings that have 3 as value</td>
<td>30% of all ratings</td>
</tr>
<tr>
<td></td>
<td>Four</td>
<td>The number of ratings that have 4 as value</td>
<td>20% of all ratings</td>
</tr>
<tr>
<td></td>
<td>Five</td>
<td>The number of ratings that have 5 as value</td>
<td>20% of all ratings</td>
</tr>
<tr>
<td>Ratings Partition</td>
<td><em>healthRelevant</em></td>
<td>The number of documents each user will rate that are relevant to some health problem he/she suffers from</td>
<td>20% of ratings from each user</td>
</tr>
<tr>
<td></td>
<td><em>nonRelevant</em></td>
<td>The number of documents that each user will rate that are not relevant to any of his/her health problems</td>
<td>80% of ratings from each user</td>
</tr>
</tbody>
</table>
Evaluation
Similarity Evaluation Measures

For evaluation we use the following measures:

Mean Absolute Error (MAE):

\[
MAE = \frac{1}{n} \sum_{i=1}^{n} |predicted_i - actual_i|
\]

Root Mean Square Error (RMSE)

\[
RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (predicted_i - actual_i)^2}
\]
RatS vs SemS (1/2)

Tests done for 100 different users
RatS vs SemS (2/2)

Test done for 100 different users
Aggregation Evaluation Measures

In order to evaluate the different aggregation methods, we compare the members individual recommendation lists with that of the group’s.

We utilize two distances:

**Spearman’s footrule distance:** is the absolute difference between the ranks assigned to an item in each list.

\[ S(t_1, t_2) = \sum_{i=1}^{n} |t_1(i) - t_2(i)| \]

**Kendall tau distance:** is a metric that counts the number of pairwise disagreements between two ranking lists

\[ K(t_1, t_2) = |\{(i, j) : i < j, t_1(i) < t_1(j) \land t_2(i) > t_2(j)\} \cup \{(i, j) : i < j, t_1(i) > t_1(j) \land t_2(i) < t_2(j)\}| \]
Kendall tau Distance

Test done for 10 different groups
Spearman footrule Distance

Test done for 10 different groups
Fairness

\[ \text{fairness}(G, D) = \frac{\sum_{u \in G} \text{fairness}(u, D)}{|G|} \]

Test done for 10 different groups
Difference between score-based and rank-based aggregation designs:

- In score-based we calculate the actual score of an item.
- In rank-based we calculate the rank of an item.

\[
value(G, D) = fairness(G, D) \cdot \sum_{i \in D} relevance(G, i)
\]
Test done for 10 different groups
Test done for 10 different groups
Test done for 10 different groups
Advantages of the method

• Propose an end-to-end framework for group recommendations
  ▫ First time in the health domain

• Incorporate fairness into recommendation process

• Introduce a novel similarity method
  ▫ Outperform rating-based similarity measures
  ▫ Exploit Semantic Information
  ▫ Able to compute users similarities without ratings

• Explore various aggregation policies and introduce the fairness aware policy
Questions ?