

**TACKLING CHRONIC DISEASE MANAGEMENT VIA COMPUTATIONAL
PHENOTYPING: ALGORITHMS, TOOLS AND APPLICATIONS**

A Dissertation
Presented to
The Academic Faculty

By

Robert Chen

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Computational Science and Engineering

Georgia Institute of Technology

August 2018

Copyright © Robert Chen 2018

TACKLING CHRONIC DISEASE MANAGEMENT VIA COMPUTATIONAL PHENOTYPING: ALGORITHMS, TOOLS AND APPLICATIONS

Approved by:

Dr. Jimeng Sun, PhD, Advisor
School of Computational Science
and Engineering
Georgia Institute of Technology

Dr. Mark Braunstein, MD
School of Interactive Computing
Georgia Institute of Technology

Dr. Jon Duke, MD
School of Computational Science
and Engineering
Georgia Institute of Technology

Dr. Shamkant Navathe, PhD
School of Computer Science
Georgia Institute of Technology

Dr. David Gutman, MD, PhD
Department of Neurology, Depart-
ment of Biomedical Informatics
Emory University

Date Approved: May 3, 2018

Data is the drug of the future. The fate of modern medicine hinges on the responsibility of computer scientists and medical professionals to embrace and accelerate innovation of quantitative methods to fight human disease.

It is incumbent upon physicians to excel in computer science. It is incumbent upon software engineers and computer scientists to excel in medicine. The future of medicine cannot be built by machine learning enthusiasts. The future of medicine will need to be built by cross-functional individuals who take the time to correctly apply reliable computational methods to real-world clinical problems in order to derive real-world impact.

Robert Chen

ACKNOWLEDGEMENTS

I would like to especially thank Jimeng Sun, Mark Braunstein, Shamkant Navathe, Jon Duke, and David Gutman for their collaboration, dedication, sacrifice and guidance throughout the research process.

Research is a team sport where everyone contributes, either large or small, to the common goal. I would like to acknowledge all countless colleagues, classmates, friends and family who helped make this work possible.

TABLE OF CONTENTS

Acknowledgments	v
List of Tables	viii
List of Figures	ix
Chapter 1: Introduction and Background	1
1.1 Phenotyping	1
1.2 Predictive Modeling and Deployment	3
1.3 Areas of Focus	4
Chapter 2: Phenotype Discovery	6
2.1 Algorithms	6
2.1.1 Rubik: Knowledge Guided Tensor Factorization and Completion for Health Data Analytics	6
2.1.2 Related Work	30
2.1.3 Conclusion	32
2.2 Applications of phenotype discovery algorithms	32
2.2.1 Segmentation of heart failure patients via nonnegative tensor fac- torization	32
Chapter 3: Predictive Modeling	41

3.1	Applications	41
3.1.1	Recurrent neural networks for early detection of heart failure from longitudinal EHR data: implications for temporal modeling with respect to time before diagnosis, data density, data quantity and data type.	41
Chapter 4: Model Deployment		63
4.1	Tools for Model Deployment	63
4.1.1	FHIR-Based Web Services for Computational Phenotyping from Electronic Health Records	63
4.1.2	Cloud-based Predictive Modeling System and its Application to Asthma Readmission Prediction	66
4.1.3	Executable Data Science Notebook for Phenotype Discovery Pipeline using Observational Data	83
Chapter 5: Discussion, Conclusions and and Closing Remarks		104
5.1	Overall Discussion and Conclusions	104
5.1.1	Creating and Capturing Value in the Healthcare Analytics Space with Phenotyping Technologies	104
5.1.2	Validation of Phenotyping Algorithms, Tools and Applications . . .	109
5.2	Closing Remarks	111
References		121

LIST OF TABLES

2.1	An example of a phenotype that a group of patients may exhibit. The diagnoses and medications are shown in rank order of importance.	8
2.2	A comparison of different tensor models	8
2.3	Notations for Rubik	11
2.4	Elements of the diagnosis and medication modes in the bias tensor.	21
2.5	An example of a Marble-derived phenotype.	23
2.6	Two subphenotypes of hypertensive patients.	23

LIST OF FIGURES

1.1	We describe several algorithms, tools and applications involved in the predictive modeling process.	2
1.2	Phenotyping refers to the automated subsetting of patients into subgroups. .	2
1.3	Each phenotype represents a persona, or a patient presentation that is meaningful and can be leveraged for future steps of the biomedical data science value chain.	3
1.4	Phenotyping adds tremendous value to the functions of all major stakeholders in the healthcare ecosystem.	4
2.1	A comparison of the meaningfulness of the phenotypes discovered by Marble and Rubik.	22
2.2	The average level of overlap in the phenotypes as a function of the pairwise constraint coefficient λ_q	25
2.3	An average similarity comparison of different methods as a function of the missing data level.	26
2.4	An average similarity comparison of different methods as a function of the noise level.	26
2.5	The similarity between the true solution and the solution under incorrect guidance as a function of the incorrect guidance level.	27
2.6	The count of non-zero elements along the three modes as a function of the thresholding parameters.	28
2.7	A runtime comparison of different methods on the Vanderbilt dataset as a function of the number of patients.	29

2.8	A runtime comparison of different methods on the CMS dataset as a function of the number of patients.	29
2.9	Proportion of contribution of each constraint.	30
2.10	Criteria for inclusion of data.	36
2.11	A graphical representation of the experimental setup.	38
2.12	Graphical representation of non-negative tensor factorization.	38
2.13	Phenotypes of patients with heart failure with reduced ejection fraction (HFrEF).	39
2.14	Phenotypes of patients with heart failure with preserved ejection fraction (HFpEF).	40
3.1	One-hot encoding of features included in model. Note: in our implementation, all 5 data domains are concatenated into single vector.	49
3.2	Graphical depiction of model architecture for heart failure prediction with GRU.	50
3.3	Prediction performance as a function of feature set used. Individual data types (demographics, medications, diagnoses, vital signs, social history) as well as combined data types were used.	52
3.4	Distribution of the number of encounters in the 2-year observation window for cases and controls.	53
3.5	Prediction performance as a function of data density. Values are recorded as AUC of prediction.	54
3.6	Prediction performance as a function of training set size. Values are recorded as AUC of prediction.	55
3.7	Prediction performance as a function of training set size, using only the vital signs as features.	56
3.8	Prediction performance as a function of training set size, using only the medications as features.	57
3.9	Prediction performance for RNN model as a function of prediction window size. Values are recorded as AUC of prediction.	57

3.10	Prediction performance for RNN model as a function of observation window size. Values are recorded as AUC of prediction.	58
4.1	An illustration of the general framework for the phenotyping web service. .	66
4.2	An overview of our predictive modeling system. We extract EHR information from the database from the hospital and store the information in these event sequence files through persistent web services running on a private dedicated server. These event sequence files are uploaded to the web service on the cloud.	71
4.3	Screenshots of the cohort construction module (A), feature construction module (B), predictive modeling module (C) and the performance analysis module (D). The cohort construction module allows users to specify criteria for selection of cases and for identifying matching controls. In the predictive modeling module, the user may specify parameters for particular feature selection and classification algorithms. The performance analysis module allows users to visualize key performance metrics as well as top predictive features selected in feature selection	73
4.4	Timeline of modules run and elapsed time. The data-splitting, training and testing times refer to the run times for each respective step of cross validation. Times are shown in seconds (s).	81
4.5	Overview of the workflow and phenotyping pipeline components	87
4.6	Screenshots from the phenotyping pipeline notebook. Shown here are specifications for a connection from the lab notebook to a database that follows the OMOP Common Data Model.	87
4.7	Screenshots from the phenotyping pipeline notebook. Shown here are excerpts from the code for running the non-negative tensor factorization based phenotyping algorithm. Descriptions of the phenotypes are displayed inline. Display of general statistics of the phenotype results are displayed in bar-chart format inline (with Matplotlib).	88
4.8	Screenshots from the phenotyping pipeline notebook. Shown here are excerpts of code for follow-up analysis that can be conducted on the phenotypes. Kaplan-Meier analysis is shown with Kaplan-Meier curve and differential survival shown in bar chart format.	89

4.9	An illustration of the phenotype definitions generated from the pipeline, which are saved in a portable JSON object. Data about the particular phenotyping algorithm and configurations (e.g., parameter tuning) used, OMOP tables used, associated concept vocabularies used, and detailed descriptions of each phenotype are included. In the descriptions of each phenotype, each particular concept included in the phenotype definition is included, along with the associated concept vocabulary and code, as well as the associated weight of contribution of each concept to the final phenotype.	92
4.10	An illustration of the phenotype cohorts generated from the pipeline, which are saved in a portable JSON object.	93
4.11	An illustration of possible workflows in different use cases of the phenotyping pipeline. The specific data sources used in our use case implementations are shown in the figure. (A) in the baseline case, the phenotype discovery pipeline is used to generate phenotype definitions and cohorts (patients belonging to each phenotype), using the non-negative tensor factorization (NTF) algorithm. (B) In Case 1, a pipeline can be used to discover phenotypes from a cohort, and a notebook and saved configurations can be saved. Subsequently, a new pipeline can be launched with the saved notebook and configurations, using a different cohort. (C) In Case 2, a phenotype assignment pipeline can be launched which takes the phenotype definitions from a previous iteration of the phenotype discovery pipeline as input, along with a new dataset, and outputs assignments of patients from the new dataset into the phenotype definitions loaded. (D) In Case 3, the phenotype discovery pipeline is customized to the users preferences, which may involve swapping out a different phenotyping algorithm; in this example, K-means is used as the phenotype discovery algorithm. Note that different phenotyping algorithms and datasets can be used depending on the specific scenario.	94
4.12	An illustration of the data filtering required for cohort construction. Patients from the Sutter Health dataset who met inclusion criteria for heart failure case definition were identified. Corresponding patients from the CMS DE-SynPUF dataset who met the same criteria were subsequently identified. . .	96
4.13	An illustration of the heart failure phenotypes found in the Sutter Health dataset. For each phenotype, the number of patients (% of cohort) and relative weight of contribution of each feature are shown.	99
4.14	An illustration of the heart failure phenotypes found in the CMS SynPUF dataset. For each phenotype, the number of patients (% of cohort) and relative weight of contribution of each feature are shown.	100

5.1	There has been a rise in the number of accountable care organizations in the United States for the past 6 years. This trend is likely to continue in the future amid a shifting regulatory environment as the United States healthcare system shifts from a service-based reimbursement model an outcome-based and cost-based reimbursement model.	106
5.2	An example of a Kaplan-Meier plot. In this particular implementation, two subgroups of patients were tracked over time with respect to survival rates. Phenotyping algorithms may be validated with survival analysis, as they may potentially identify patient subgroups that exhibit differential response to therapy.	108
5.3	Sample analysis plan for a cost comparison analysis of phenotypes derived from a phenotyping algorithm.	109
5.4	Cardiovascular patients undergoing cardiac imaging (cardiac MRI, echocardiogram) were automatically grouped into 10 phenotypes, which show vast differences in survival after over a decade of follow-up.	111

SUMMARY

With the recent tsunami of medical data from electronic health records (EHRs), there has been a rise in interest in leveraging such data to improve efficiency of healthcare delivery and improve clinical outcomes. A large part of medical data science involves computational phenotyping, which leverages data driven methods to subtype and characterize patient conditions from heterogeneous EHR data. While many applications have used supervised phenotyping, unsupervised phenotyping will become increasingly more important in future precision medicine initiatives. A typical healthcare analytics workflow consists of phenotype discovery from EHR data, followed by predictive modeling that may leverage such phenotypes, followed by model deployment via avenues such as the Fast Interoperable Healthcare Interoperability Resources (FHIR) specification. To address unmet clinical needs, we have developed and demonstrated algorithms, tools and applications along each step of this process. In the area of unsupervised phenotyping, we developed novel tensor factorization based methods for discovery of phenotypes from EHR data. Further, we proposed a portable, open source data science notebook to facilitate phenotype discovery and sharing of methods and results across research institutions. Finally, we demonstrate the application of unsupervised phenotyping based on tensor factorization in phenotype discovery among heart failure patients. In the area of predictive modeling, we demonstrate a novel deep learning algorithm and framework for early detection of heart failure from cohorts of over 30,000 patients from large health systems. In the area of model deployment, we propose several tools for discovering high-risk patients in patient cohorts, ranging from months to years before potential unfavorable events. We propose large-scale deployment of predictive modeling for health systems via deployment of machine learning strategies on the Amazon Web Services Elastic MapReduce cluster, and aim to demonstrate the utility on tasks such as readmission prediction on cohorts of up to 2 million patients. Furthermore, we propose a system for deployment of phenotyping algorithms on Amazon Web Services

using the HL7 FHIR data model, and demonstrate the utility in an application for phenotyping of cohorts of over 30,000 intensive care patients in the inpatient setting.

CHAPTER 1

INTRODUCTION AND BACKGROUND

A typical healthcare analytics workflow consists of phenotype discovery from EHR data, followed by predictive modeling that may leverage such phenotypes, followed by model deployment via avenues such as FHIR. While many clinical researchers, clinicians, and medical data scientists aspire to perform data science today, there exist major hurdles around development of algorithms, tools, and applications across all areas of the typical workflow. In this thesis, we identified major unmet needs, designed and implemented solutions in these areas across the entire medical data science value chain, and propose several avenues for additional work.

A typical data science value chain is shown in Figure 1.1. Note that typically we begin with EHR data and feed it through a series of modules: phenotype discovery, predictive modeling, and model deployment. Each of these broad modules, relies on algorithms, tools and applications for successful data science execution.

1.1 Phenotyping

Phenotyping generally refers to extraction of meaningful clinical concepts from medical data. A phenotype can be thought of as a set of features that describe a general patient state or presentation. For example, Figure 1.2 illustrates the high level demonstration of extracting 3 phenotypes from a cohort of patients. Two examples of phenotypes are shown, one (Phenotype 1) which describes patients who have a variety of mild conditions, and another (Phenotype 2) which describes patients that have a set of more severe conditions.

It should be noted that while phenotypes can be described in terms of collections of features (in the example above, these features are diseases), a phenotypes features each have an associated weight, which describes the importance or contribution of the feature to

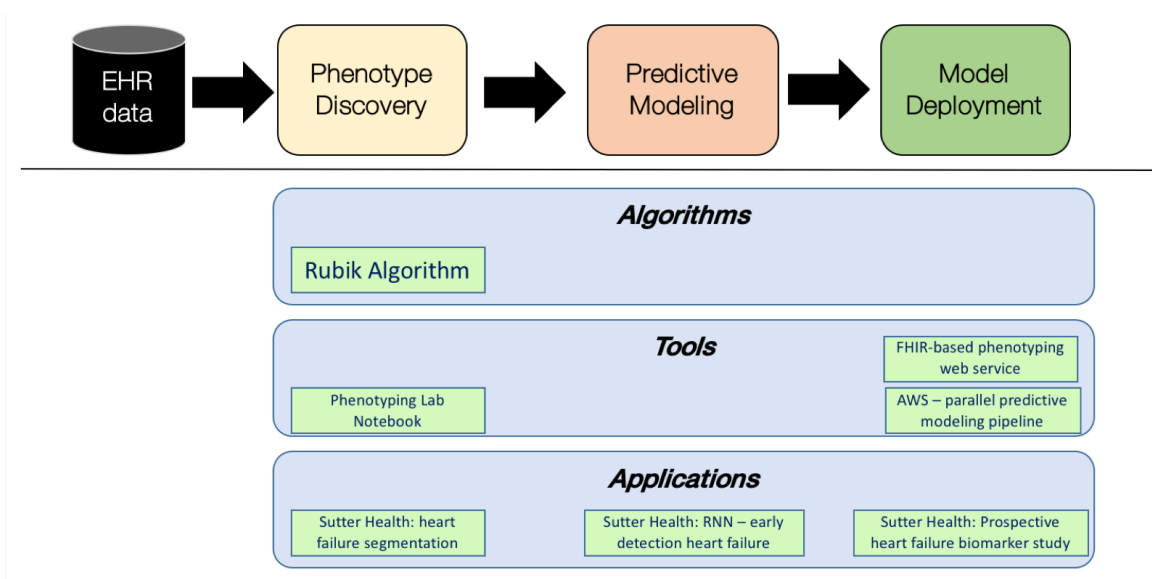


Figure 1.1: We describe several algorithms, tools and applications involved in the predictive modeling process.

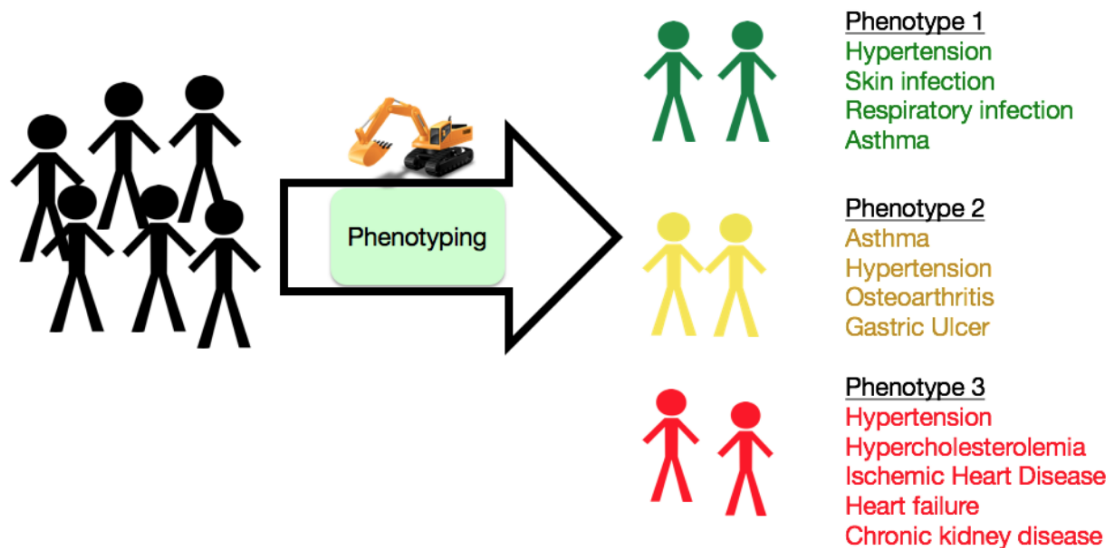


Figure 1.2: Phenotyping refers to the automated subsetting of patients into subgroups.

the overall phenotype. In the example shown in the Figure 1.3 below, two phenotypes are shown, each of which is composed of sets of diagnoses (shown in blue) and medications (shown in green). Note that all diagnoses have weights which sum to 1 and all medications have weights that sum to 1.

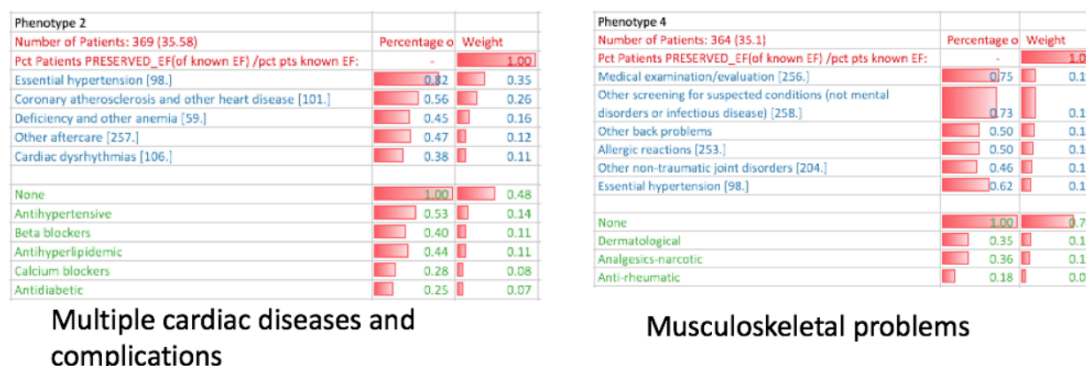


Figure 1.3: Each phenotype represents a persona, or a patient presentation that is meaningful and can be leveraged for future steps of the biomedical data science value chain.

Phenotyping is useful because it allows all major stakeholders in the healthcare system to gain actionable insights in order to motivate care delivery (Figure 1.4).

1.2 Predictive Modeling and Deployment

Predictive modeling can be leveraged for applications such as early detection of chronic diseases. While predictive modeling is broad in nature, in our work we refer to predictive modeling as usage of machine learning classification methods using features constructed from medical data, namely EHR data. Predictive features used can include phenotypes discovered in the phenotype discovery process as well.

Model deployment refers to the usage of phenotype discovery and predictive modeling methods to directly guide prospective research or care coordination.

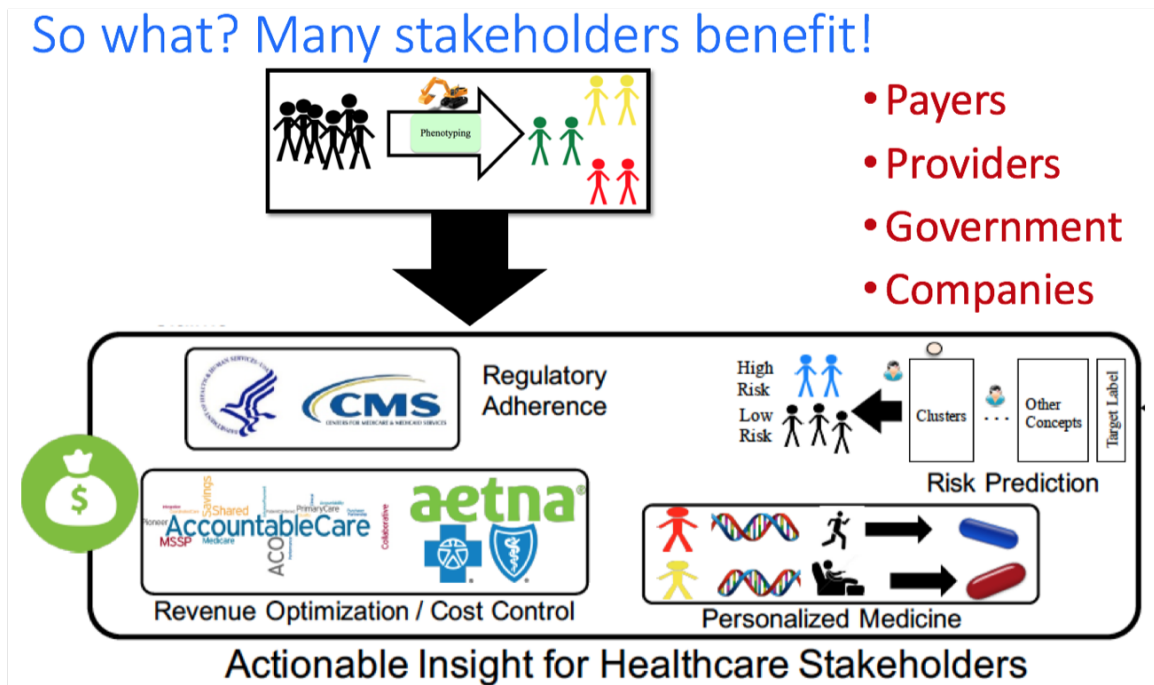


Figure 1.4: Phenotyping adds tremendous value to the functions of all major stakeholders in the healthcare ecosystem.

1.3 Areas of Focus

In this thesis we focus on several areas: phenotype discovery, predictive modeling, and model deployment.

In the area of unsupervised phenotyping, we developed novel tensor factorization based methods for discovery of phenotypes from EHR data. Further, we proposed a portable, open source data science notebook to facilitate phenotype discovery and sharing of methods and results across research institutions. Finally, we demonstrate the application of unsupervised phenotyping based on tensor factorization in phenotype discovery among heart failure patients. In the area of predictive modeling, we demonstrate a novel deep learning algorithm and framework for early detection of heart failure from patient cohorts of over 30,000 patients from large health systems.

In the area of model deployment, we propose several tools for discovering high-risk patients in patient cohorts, ranging from months to years before potential unfavorable events.

We propose large-scale deployment of predictive modeling for health systems via deployment of machine learning strategies on the Amazon Web Services Elastic MapReduce cluster, and aim to demonstrate the utility on tasks such as readmission prediction on cohorts of up to 2 million patients. Furthermore, we propose a system for deployment of phenotyping algorithms on Amazon Web Services using the HL7 FHIR data model, and demonstrate the utility in an application for phenotyping of cohorts of over 30,000 intensive care patients in the inpatient setting.

CHAPTER 2

PHENOTYPE DISCOVERY

In this chapter we describe methods for uncovering phenotypic subsets of patients from heterogeneous electronic health record data.

2.1 Algorithms

We describe Rubik, a phenotyping algorithm based on nonnegative tensor factorization.

2.1.1 Rubik: Knowledge Guided Tensor Factorization and Completion for Health Data Analytics

Abstract

Computational phenotyping is the process of converting heterogeneous electronic health records (EHRs) into meaningful clinical concepts. Unsupervised phenotyping methods have the potential to leverage a vast amount of labeled EHR data for phenotype discovery. However, existing unsupervised phenotyping methods do not incorporate current medical knowledge and cannot directly handle missing or noisy data. We propose Rubik, a constrained, non-negative tensor factorization and completion method for phenotyping. Rubik incorporates 1) guidance constraints to align with existing medical knowledge, and 2) pairwise constraints for obtaining distinct, non-overlapping phenotypes. Rubik also has built-in tensor completion that can significantly alleviate the impact of noisy and missing data. We utilize the Alternating Direction Method of Multipliers (ADMM) framework to tensor factorization and completion, which can be easily scaled through parallel computing. We evaluate Rubik on two EHR datasets, one of which contains 647,118 records for 7,744 patients from an outpatient clinic, the other of which is a public dataset containing

1,018,614 CMS claims records for 472,645 patients. Our results show that Rubik can discover more meaningful and distinct phenotypes than the baselines. In particular, by using knowledge guidance constraints, Rubik can also discover sub-phenotypes for several major diseases. Rubik also runs around seven times faster than current state-of-the-art tensor methods. Finally, Rubik is scalable to large datasets containing millions of EHR records.

Introduction

The widespread adoption of EHR systems in the United States and many other countries has resulted in a tsunami of EHR data, which is becoming an increasingly important source of detailed medical information. Successful phenotyping efforts on EHR data can enable many important applications, such as clinical predictive modeling [1, 2] and EHR-based genomic association studies [1, 3, 2]. Furthermore, medical professionals are accustomed to reasoning based on concise and meaningful phenotypes. Thus, it is imperative that robust phenotyping methods be developed and refined to keep up with the growing volume and heterogeneity of EHR data.

A typical phenotyping algorithm takes EHR data as input, and defines a group or several groups of patients, each of which is referred to as a phenotype. An example of a phenotype is shown in Table 2.1, which depicts a collection of diseases and associated medications that may co-occur in a patient¹. In Table 2.1 and in all subsequent displays of phenotypes throughout the paper, the diagnoses and medications are shown by rank order of importance.

Most existing phenotyping methods are supervised approaches, which are either expert-defined rule-based methods [10] or classification methods [11]. However, as labeled data are difficult to obtain, efficient unsupervised phenotyping approaches are needed to leverage the vast amount of unlabeled EHR data for discovering multiple interconnected phenotypes. The only such algorithm to our knowledge is based on sparse nonnegative tensor

¹Note that individuals that belong to a phenotype will often have some, but not all, of the diagnoses and medications listed in the phenotype definition.

Table 2.1: An example of a phenotype that a group of patients may exhibit. The diagnoses and medications are shown in rank order of importance.

Diagnoses	Medications
Hypertension	Statins
Ischemic heart disease	Angiotensin receptor blockers
Hyperlipidemia	ACE inhibitors
Obesity	Loop diuretics
	Cardioselective beta blockers

Table 2.2: A comparison of different tensor models

Property	Marble [4]	FaLRTC [5]	CTMF [6]	TF-BPP [7]	WCP [8]	NETWORK [9]	Rubik
Factorization	✓		✓	✓	✓	✓	✓
Completion		✓			✓		✓
Non-negativity	✓			✓			✓
Guidance						✓	✓
Sparse solution	✓						✓
Scalability							✓

factorization [12, 4], which models interconnected data as tensors and discovers sparse nonnegative factors as phenotypes. However, there are still several formidable challenges in unsupervised phenotyping methods:

- **Leverage of existing knowledge.** Existing medical knowledge, such as a physician’s experience or a medical ontology, should be incorporated into the phenotyping algorithms in order to identify more meaningful phenotypes that align more closely with existing medical knowledge.
- **Deriving distinct phenotypes.** Existing unsupervised phenotyping methods, such as Marble [4], can lead to overlapping phenotypes which hinder their interpretation. Ideally, the resulting phenotypes should be distinct without much overlap.
- **Missing and noisy data.** EHR data often contain missing and noisy records. It is important to ensure that phenotyping algorithms remain robust against missing and noisy data.
- **Scalability.** Real world EHR data contains millions of records and spans multiple dimensions. It is important to develop innovative phenotyping methods that scale well with increases in data size.

Our contributions: We propose *Rubik*, an unsupervised phenotyping method based on tensor factorization and completion, which addresses all the aforementioned challenges:

- We incorporate guidance constraints based upon medical knowledge in order to derive clinically meaningful phenotypes.
- We introduce pairwise constraints in the formulation to ensure distinct phenotypes.
- Our proposed algorithm embeds efficient tensor completion, thereby alleviating both missing and noisy information in EHR tensors.
- We design a scalable algorithm based on Alternating Direction Method of Multipliers (ADMM) for solving this problem, which significantly outperforms several baseline methods.

We evaluate Rubik on two large EHR datasets. Our results demonstrate that Rubik achieves at least a 60% reduction in the number of overlapping phenotypes compared to Marble as a baseline [4]. Rubik also increases the number of meaningful phenotypes by 50%. Furthermore, the phenotypes and the baseline characteristics derived from the real EHR data are consistent with existing studies on the population. Rubik is also much more computationally scalable compared to all baseline methods, with up to a 7-fold decrease in running time over the baselines.

Table 2.2 compares the properties between Rubik and other tensor methods.

Outline: The remainder of the paper is organized as follows. We review preliminaries in Sec. 2. We present our framework in Sec. 3. Datasets and experimental evaluation are discussed in Sec. 4. Related work is summarized in Sec. 5. Finally, we conclude by discussing future research directions.

RUBIK

We first formulate the problem and then provide a general overview of the formulation. Finally, we present an efficient optimization algorithm for solving the problem.

Formulation

We formulate our model as a constrained tensor optimization, where four constraints (one hard and three soft) are involved:

- **Completion:** This is the hard constraint. The unknown full tensor \mathcal{X} matches the observed elements in the partially observed tensor \mathcal{O} .
- **Guidance:** A subset of columns in a factor matrix $\mathbf{A}^{(p)}$ are close to the columns represented by prior knowledge $\hat{\mathbf{A}}^{(p)}$.
- **Pairwise constraints:** The columns in a factor matrix $\mathbf{A}^{(k)}$ should be close to orthogonal.
- **Non-negativity:** The factor matrices $\mathbf{A}^{(1)}, \dots, \mathbf{A}^{(N)}$ contain only nonnegative entries to enhance interpretability.
- **Sparsity:** We adjust the sparsity of each factor matrix $\mathbf{A}^{(n)}$ by removing non-zero entries less than γ_n .

$$\min_{\mathcal{X}, \mathcal{T}, \mathcal{C}} \left\{ \Phi(\mathcal{X}, \mathcal{T}, \mathcal{C}) \right\}, \quad \text{s.t. } \underbrace{\mathcal{P}_\Omega(\mathcal{X}) = \mathcal{P}_\Omega(\mathcal{O})}_{\text{Completion}}$$

where,

$$\begin{aligned} \Phi = & \underbrace{\|\mathcal{X} - \mathcal{C} - \mathcal{T}\|_F^2}_{\text{Factorization error}} + \frac{\lambda_a}{2} \underbrace{\|(\mathbf{A}^{(p)} - \hat{\mathbf{A}}^{(p)})\mathbf{W}\|_F^2}_{\text{Guidance information}} \\ & + \frac{\lambda_q}{2} \underbrace{\|\mathbf{Q} - \mathbf{A}^{(k)T} \mathbf{A}^{(k)}\|_F^2}_{\text{Pairwise constraint}} \\ \mathcal{T} = & \underbrace{[\![\mathbf{A}^{(1)}; \dots; \mathbf{A}^{(N)}]\!] \in \Omega_{\mathcal{T}}}_{\text{Interaction tensor}}, \quad \mathcal{C} = \underbrace{[\![\mathbf{u}^{(1)}; \dots; \mathbf{u}^{(N)}]\!] \in \Omega_{\mathcal{C}}}_{\text{Bias tensor}} \\ \Omega_{\mathcal{T}} = & \Omega_{A_1} \times \dots \times \Omega_{A_N}, \quad \Omega_{A_n} = \underbrace{\{\mathbf{A} \in \{0\} \cup [\gamma_n, +\infty)^{I_n \times R}\}}_{\text{Sparse representation}} \\ \Omega_{\mathcal{C}} = & \Omega_{u_1} \times \dots \times \Omega_{u_N}, \quad \Omega_{u_n} = \{\mathbf{u} \in [0, +\infty)^{I_n \times 1}\} \end{aligned} \tag{2.1}$$

Next, we formally define all the necessary notations in Table 2.3.

²A rank- R tensor is defined as the sum of R rank-one tensor

Table 2.3: Notations for Rubik

Notation	Definition
$\mathcal{X} \in \mathcal{R}_+^{I_1 \times \dots \times I_N}$	unknown full tensor
$\mathcal{O} \in \mathcal{R}_+^{I_1 \times \dots \times I_N}$	partially observed tensor
Ω	the set of observed indices
\mathcal{P}_Ω	set all but elements in Ω to zero
$\mathcal{X} \approx \mathcal{C} + \mathcal{T}$	\mathcal{C} : rank-one bias tensor \mathcal{T} : rank- R interaction tensor ²
$\hat{\mathbf{A}}^{(p)} \in \mathcal{R}_+^{I_p \times R}$	guidance matrix on mode- p
$\mathbf{Q} \in \mathcal{R}_+^{R \times R}$	pairwise constraint matrix
$\mathbf{W} \in \mathcal{R}_+^{R \times R}$	weight matrix for guidance constraints

In particular, the unknown full tensor \mathcal{X} is approximated by two terms, 1) a rank one bias tensor \mathcal{C} and 2) a rank- R interaction tensor \mathcal{T} . The bias tensor \mathcal{C} captures the baseline characteristics of the entire tensor, which is a rank-one tensor with all positive vectors $\llbracket \mathbf{u}^{(1)}, \dots, \mathbf{u}^{(N)} \rrbracket$. The interaction tensor \mathcal{T} is a CP tensor model, with nonnegative constraints on all factor matrices. Let $\mathbf{Q} \in \mathcal{R}_+^{R \times R}$ denote the pairwise constraints for a specific factor matrix say $\mathbf{A}^{(k)}$ with positive λ_q capturing the weights of this constraint. In our experiments, we set \mathbf{Q} to be an identity matrix. The guidance knowledge is encoded as a vector where positive entries indicate relevant feature dimensions. For example, we can have a guidance vector corresponding to a hypertension diagnosis, where the hypertension-related entries are set to positive values (e.g., one), and the remaining entries are zero. Now, let us assume that we have R' guidance vectors ($R' \leq R$). To ease subsequent derivation, we construct the guidance matrix $\hat{\mathbf{A}}^{(p)}$ by adding zero columns to make $\hat{\mathbf{A}}^{(p)}$ of the same size as the corresponding factor matrix $\mathbf{A}^{(p)} \in \mathcal{R}_+^{I_p \times R}$. Then, to ignore the effects of those zero columns, we multiply the difference between $\mathbf{A}^{(p)}$ and $\hat{\mathbf{A}}^{(p)}$ by a weight matrix

$$\mathbf{W} = \begin{pmatrix} I & 0 \\ 0 & 0 \end{pmatrix}$$

where $I \in \mathcal{R}^{R' \times R'}$ is the identity matrix.

Next, we explain the intuition behind the model.

Problem Overview

At a high-level, Rubik aims to simultaneously conduct the non-negative CP factorization and recover the non-negative low rank tensor \mathcal{X} from a *partially observed tensor* \mathcal{O} . This idea is captured through the *Factorization error* and *Completion* term in Eq. 2.1. Note that each $\mathbf{A}^{(k)} \in \mathcal{R}_+^{I_k \times R}$ has an estimated rank of R . Hence the interaction tensor \mathcal{T} has rank up to R and the low rank property is enforced.

Bias tensor. Rubik includes a rank-one bias tensor \mathcal{C} to capture the baseline characteristics common in the overall population, which is similar to Marble [4]. In phenotyping applications, it represents the common characteristics of the n^{th} phenotype amongst the entire population (e.g., the value of the element in the diagnosis mode of the bias tensor corresponding to hypertension represents the overall possibility of any given patient having hypertension).

Guidance information. In real world applications, we may know guidance information that can be encoded into the corresponding factor matrices. For example, we might have the knowledge that some phenotypes should be related to hypertension. Utilizing this guidance information can lead to more intuitive and understandable phenotypes. This constraint is captured through the *Guidance information* term in Eq. 2.1.

Pairwise constraints. We hope to discover distinct phenotypes in order to obtain more concise and interpretable results. We can penalize the cases where phenotypes have overlapping dimensions (e.g., two common diagnoses between two phenotype candidates from the diagnosis mode). This constraint is captured through the *Pairwise constraint* term in Eq. 2.1.

Algorithm

Next we describe the detailed algorithm. The main idea is to decouple constraints using an Alternating Direction Method of Multipliers (ADMM) scheme [13]. For each mode, the algorithm first computes the factor matrix associated with the interaction tensor. Once the interaction factor matrix is computed, the bias vector is computed. The whole process is repeated until convergence occurs.

Convex Subproblem

Originally, the objective function Φ is non-convex with respect to $\mathbf{A}^{(k)}$ due to the fourth order term in pairwise constraints $\|\mathbf{Q} - \mathbf{A}^{(k)T} \mathbf{A}^{(k)}\|_F^2$. By variable substitution $\mathbf{A}^{(k)} = \mathbf{B}^{(k)}$ in $\|\mathbf{Q} - \mathbf{A}^{(k)T} \mathbf{A}^{(k)}\|_F^2$ from Eq. 2.1, we obtain the equivalent form of Φ

$$\begin{aligned} \Psi = & \|\mathcal{X} - \mathcal{C} - \mathcal{T}\|_F^2 + \frac{\lambda_a}{2} \|(\mathbf{A}^{(p)} - \hat{\mathbf{A}}^{(p)}) \mathbf{W}\|_F^2 \\ & + \frac{\lambda_q}{2} \|\mathbf{Q} - \mathbf{B}^{(k)T} \mathbf{A}^{(k)}\|_F^2 \end{aligned}$$

Note that the objective function Ψ is now convex w.r.t. $\mathbf{A}^{(k)}$.

Using a similar variable substitution technique, Eq. 2.1 is reformulated into the following equivalent form:

$$\begin{aligned} & \min_{\mathcal{X}, \mathcal{T}, \mathcal{C}, \mathcal{B}, \mathcal{V}} \{ \Psi \} \tag{2.2} \\ \text{s.t. } & \mathbf{B}^{(k)} = \mathbf{A}^{(k)}, \mathbf{B}^{(k)} \in \Omega_{An}, n = 1, \dots, N \\ & \mathbf{v}^{(n)} = \mathbf{u}^{(n)}, \mathbf{v}^{(n)} \in \Omega_{un}, n = 1, \dots, N \\ & \mathcal{P}_\Omega(\mathcal{X}) = \mathcal{P}_\Omega(\mathcal{O}) \end{aligned}$$

where $\mathcal{B} = \{\mathbf{B}^{(1)}, \dots, \mathbf{B}^{(N)}\}$ and $\mathcal{V} = \{\mathbf{v}^{(1)}, \dots, \mathbf{v}^{(N)}\}$ are the collections of auxiliary variables.

Solving Scheme

The partial augmented Lagrangian function for Ψ is:

$$\begin{aligned} \mathcal{L} = \Psi &+ \sum_{n=1}^N (\langle \mathbf{p}^{(n)}, \mathbf{v}^{(n)} - \mathbf{u}^{(n)} \rangle + \frac{\eta}{2} \|\mathbf{v}^{(n)} - \mathbf{u}^{(n)}\|_F^2) \\ &+ \sum_{n=1}^N (\langle \mathbf{Y}^{(n)}, \mathbf{B}^{(n)} - \mathbf{A}^{(n)} \rangle + \frac{\mu}{2} \|\mathbf{B}^{(n)} - \mathbf{A}^{(n)}\|_F^2) \end{aligned} \quad (2.3)$$

where $\mathcal{Y} = \{\mathbf{Y}^{(1)}, \dots, \mathbf{Y}^{(N)}\}$ and $\mathcal{P} = \{\mathbf{p}^{(1)}, \dots, \mathbf{p}^{(N)}\}$ are the set of Lagrange multipliers. $\langle X, Y \rangle = \sum_{ij} X_{ij} Y_{ij}$ denotes the inner product of two matrices X and Y . $\{\eta, \mu\}$ are penalty parameters, which can be adjusted efficiently according to [14].

Here we solve Eq. 2.3 by successively minimizing the Lagrangian with respect to each variable in block coordinate descent procedures. Each iteration involves updating one variable, with the other variables fixed to their most recent values. The updating rules are as follows.

Update the interaction tensor. Without loss of generality, we assume that the prior and pairwise guidance information are on the n^{th} mode. One can easily modify λ_a, λ_q to zero if there is no guidance information on a particular mode. Set $\mathcal{R} = \mathcal{X} - \mathcal{C}$, which is the residual tensor left over after subtracting the effects of the bias tensor in objective function Ψ . To compute $\mathbf{A}_{t+1}^{(n)}$, the smooth optimization problem is formulated as follows:

$$\begin{aligned} \min_{\mathbf{A}^{(n)}} & \|\mathbf{A}^{(n)} \mathbf{\Pi}^{(n)T} - \mathbf{R}_{(n)}\|_F^2 \\ &+ \frac{\lambda_q}{2} \|\mathbf{Q} - \mathbf{B}_t^{(n)T} \mathbf{A}^{(n)}\|_F^2 + \frac{\lambda_a}{2} \|(\mathbf{A}^{(n)} - \hat{\mathbf{A}}^{(n)}) \mathbf{W}\|_F^2 \\ &+ \frac{\mu_t}{2} \|\mathbf{A}^{(n)} - \mathbf{B}_t^{(n)} - \mathbf{Y}_t^{(n)} / \mu_t\|_F^2 \end{aligned} \quad (2.4)$$

where $\Pi^{(n)}$ is defined as

$$\Pi^{(n)} = \left(\mathbf{A}_t^{(N)} \odot \cdots \odot \mathbf{A}_t^{(n+1)} \odot \mathbf{A}_{t+1}^{(n-1)} \odot \cdots \odot \mathbf{A}_{t+1}^{(1)} \right)$$

The update rule of $\mathbf{A}^{(n)}$ exists in closed form. Next, we set the derivatives of Eq. 2.4 with respect to $\mathbf{A}^{(n)}$ to zero. The closed form solution thus is:

$$\begin{aligned} \mathbf{A}_{t+1}^{(n)} = & \left\{ 2\mathbf{R}_{(n)}\Pi^{(n)} + \lambda_a \hat{\mathbf{A}}^{(n)} \mathbf{W} \right. \\ & \left. + \lambda_q (\mathbf{B}_t^{(n)} (\mathbf{B}_t^{(n)})^T)^{-1} \mathbf{B}_t^{(n)} \mathbf{Q} + \mu_t \mathbf{B}_t^{(n)} + \mathbf{Y}_t^{(n)} \right\} \\ & \left\{ 2(\Pi^{(n)})^T \Pi^{(n)} + \lambda_a \mathbf{W} + (\lambda_q + \mu_t) \mathbf{I} \right\}^{-1} \end{aligned} \quad (2.5)$$

To solve for the auxiliary variable $\mathbf{B}^{(n)}$, we obtain the following optimization problem.

$$\min_{\mathbf{B}^{(n)} \in \Omega_{An}} \langle \mathbf{Y}_t^{(n)}, \mathbf{A}_{t+1}^{(n)} - \mathbf{B}^{(n)} \rangle + \frac{\mu_t}{2} \|\mathbf{B}^{(n)} - \mathbf{A}_{t+1}^{(n)}\|_F^2 \quad (2.6)$$

The closed form update for $\mathbf{B}^{(n)}$ is:

$$\mathbf{B}_{t+1}^{(n)} = \begin{cases} \mathbf{A}_{t+1}^{(n)} + \frac{1}{\mu_t} \mathbf{Y}_t^{(n)} & \text{if } \gamma_n \leq \mathbf{A}_{t+1}^{(n)} + \frac{1}{\mu_t} \mathbf{Y}_t^{(n)} \\ 0 & \text{otherwise} \end{cases} \quad (2.7)$$

Efficient computation of $(\Pi^{(n)})^T \Pi^{(n)}$. For two matrices \mathbf{M}, \mathbf{N} , we have the following property of the Khatri-Rao product [15]:

$$(\mathbf{M} \odot \mathbf{N})^T (\mathbf{M} \odot \mathbf{N}) = \mathbf{M}^T \mathbf{M} * \mathbf{N}^T \mathbf{N}$$

As a result, we can efficiently compute $(\Pi^{(n)})^T \Pi^{(n)}$ as

$$(\Pi^{(n)})^T \Pi^{(n)} = \left(\mathbf{V}_t^{(N)} * \cdots * \mathbf{V}_t^{(n+1)} * \mathbf{V}_{t+1}^{(n-1)} * \cdots * \mathbf{V}_{t+1}^{(1)} \right)$$

where $\mathbf{V}^{(m)} = (\mathbf{A}^{(m)})^T \mathbf{A}^{(m)} \in \mathcal{R}^{k \times k}$ for all $m \neq n$.

Update the bias tensor. At this point, we set \mathcal{E} to be the residual tensor left over after subtracting the effects of interaction tensor in objective function Φ . $\mathcal{E} = \mathcal{X} - \mathcal{T}$. For each $\mathbf{u}^{(n)}$, we solve the following problem,

$$\min_{\mathbf{u}^{(n)}} \|\mathbf{u}^{(n)} (\mathbf{\Lambda}^{(n)})^T - \mathbf{E}_{(n)}\|_F^2 + \frac{\eta_t}{2} \|\mathbf{u}^{(n)} - \mathbf{v}^{(n)} - \mathbf{p}^{(n)} / \eta_t\|_F^2$$

where $\mathbf{E}_{(n)}$ is the mode- n matricization of tensor \mathcal{E} . $\mathbf{\Lambda}^{(n)}$ is defined as:

$$\mathbf{\Lambda}^{(n)} = \left(\mathbf{u}_t^{(N)} \odot \dots \odot \mathbf{u}_t^{(n+1)} \odot \mathbf{u}_{t+1}^{(n-1)} \odot \dots \odot \mathbf{u}_{t+1}^{(1)} \right)$$

The closed form solution for $\mathbf{u}^{(n)}$ is

$$\mathbf{u}_{t+1}^{(n)} = \{2\mathbf{E}_{(n)}\mathbf{\Lambda}^{(n)} + \eta_t \mathbf{v}^{(n)} + \mathbf{p}_t^{(n)}\} \{2(\mathbf{\Lambda}^{(n)})^T \mathbf{\Lambda}^{(n)} + \eta_t \mathbf{I}\}^{-1} \quad (2.8)$$

The optimization problem for auxiliary variable $\mathbf{v}^{(n)}$ is:

$$\min_{\mathbf{v}^{(n)} \succeq 0} \langle \mathbf{p}^{(n)}, \mathbf{u}^{(n)} - \mathbf{v}^{(n)} \rangle + \frac{\eta_t}{2} \|\mathbf{u}^{(n)} - \mathbf{v}^{(n)}\|_F^2$$

The closed form solution is

$$\mathbf{v}_{t+1}^{(n)} = \max(0, \mathbf{u}_{t+1}^{(n)} + \frac{1}{\eta_t} \mathbf{p}_t^{(n)}) \quad (2.9)$$

Update Lagrange multipliers. We optimize the Lagrange multipliers using gradient ascent. $\mathbf{Y}_{t+1}^{(n)}, \mathbf{p}_{t+1}^{(n)}$ can be directly updated by

$$\mathbf{Y}_{t+1}^{(n)} = \mathbf{Y}_t^{(n)} + \mu_t (\mathbf{B}_{t+1}^{(n)} - \mathbf{A}_{t+1}^{(n)}) \quad (2.10)$$

Algorithm 1 MINIMIZE Ψ

```
1: Input:  $\mathcal{O}, \hat{\mathbf{A}}, \mathbf{W}, \mathbf{Q}, \lambda_a, \lambda_q$ 
2: Initialize  $\mathbf{A}_0^{(n)}$  randomly, set  $\mathbf{Y}_0^{(n)} = \mathbf{0}, \mathbf{p}_0^{(n)} = \mathbf{0}, n \in \{1, \dots, N\}, \mu_0 = 10^{-7},$   
    $\eta_0 = 10^{-7}, \mu_{max} = 10^{15}, \eta_{max} = 10^{11}, \rho = 1.05.$ 
3: repeat
4:   for  $n=1:N$  do
5:     Update  $\mathbf{A}_{t+1}^{(n)}$  and  $\mathbf{B}_{t+1}^{(n)}$  by Eq. 2.4 and Eq. 2.7
6:     Update  $\mathbf{u}_{t+1}^{(n)}$  and  $\mathbf{v}_{t+1}^{(n)}$  by Eq. 2.8 and Eq. 2.9
7:     Update  $\mathbf{Y}_{t+1}^{(n)}$  and  $\mathbf{p}_{t+1}^{(n)}$  by Eq. 2.10 and Eq. 2.11
8:   end for
9:   Update  $\mathcal{X}_{t+1}$  by Eq. 2.12
10:  Update parameter  $\mu_{t+1}$  by  $\mu_{t+1} = \min(\rho\mu_t, \mu_{max})$ 
11:  Update parameter  $\eta_{t+1}$  by  $\eta_{t+1} = \min(\rho\eta_t, \eta_{max})$ 
12: until  $\max\{\|\mathbf{A}_{t+1}^{(n)} - \mathbf{B}_{t+1}^{(n)}\|_F, n \in \{1, \dots, N\}\} \leq \epsilon$ 
13: return  $\mathcal{T}, \mathcal{C}$ 
```

$$\mathbf{p}_{t+1}^{(n)} = \mathbf{p}_t^{(n)} + \eta_t(\mathbf{v}_{t+1}^{(n)} - \mathbf{u}_{t+1}^{(n)}) \quad (2.11)$$

Update the full tensor. We now have the following optimization problem w.r.t. \mathcal{X} :

$$\min_{\mathcal{X}} \|\mathcal{X} - \mathcal{T}_{t+1} - \mathcal{C}_{t+1}\|_F^2 \quad \text{s.t., } \mathcal{P}_{\Omega}(\mathcal{X}) = \mathcal{P}_{\Omega}(\mathcal{O})$$

The optimal solution is

$$\mathcal{X}_{t+1} = \mathcal{P}_{\Omega^c}(\mathcal{T}_{t+1} + \mathcal{C}_{t+1}) + \mathcal{P}_{\Omega}(\mathcal{O}) \quad (2.12)$$

where Ω^c is the complement of Ω , i.e. the set of indexes of the unobserved entries.

Based on the above analysis, we develop the ADMM scheme for Rubik, as described in Algorithm 1.

Analysis and Discussion

Parallelization. Our scheme follows the Gauss-Seidel type of updating rule. The Jacobi type updating rules can easily be implemented with slight modification. As a result, our

algorithm can be parallelized and scaled to handle large datasets.

Flexible extension. Rubik can be easily modified to incorporate other types of guidance, depending on the domain application. For instance, in the analysis of brain fMRI data [9], we might need to consider pairwise relationships between different factor matrices. This cross-mode regularization can also be easily incorporated in our framework.

Complexity analysis. The time complexity is mainly consumed by computing $\Pi^{(n)}$ and $\mathbf{A}^{(n)}$ in Eq. 2.5, which is $O(R \sum_{i=1}^N \prod_{j \neq n}^N I_j + R \prod_{j=1}^N I_j)$. Now, let us denote the size of the largest mode as D . Then Rubik has the complexity of $O(D^N)$. Although Rubik incorporates guidance information, the computational complexity remains the same as state-of-the-art methods such as Marble [4], CP-APR [4] and WCP [8]. These competitors have similar time complexity, but they are much slower in practice due to their gradient based solving scheme and time consuming line search. By contrast, Rubik yields closed-form updates at each iteration.

Experiments

Datasets

We evaluate Rubik with two EHR datasets, from Vanderbilt University and the Center for Medicare and Medicaid Services (CMS), each of which contains diagnosis and medication information and is roughly in the form of a tensor. Raw diagnosis data in both datasets are encoded following the International Classification of Diseases (ICD-9) classification system. To avoid an overly sparse concept space, similar diagnoses codes were grouped together according to the Phenome-wide Association Study (PheWAS) terminology [16] and medications were grouped by their corresponding classes using the RxNorm ontology³.

Vanderbilt. We use a de-identified EHR dataset from Vanderbilt University Medical Center with 7,744 patients over 5 years of observation. We construct a 3^{rd} order tensor with patient, diagnosis and medication modes of size 7,744 by 1,059 by 501, respectively.

³<http://www.nlm.nih.gov/research/umls/rxnorm/>

The tensor element $\mathcal{X}(i, j, k) = 1$ if patient i is prescribed with medication k for treating diagnosis j .⁴

CMS. We used a subset of the publicly available CMS 2008-2010 Data Entrepreneurs’ Synthetic Public Use File (DE-SynPUF) dataset from the CMS [17]. For this dataset, the tensor element is based on all co-occurrences of prescription medication events and diagnoses from outpatient claims of the same patient happening on the same date, for years 2008-2010. Specifically, we constructed a tensor representing 472,645 patients by 11,424 diagnoses by 262,312 medication events.

The goal of our evaluation is four fold:

1. **Phenotype discovery:** Analyze how Rubik discovers meaningful and distinct phenotypes with different combinations of guidance.
2. **Noise analysis:** Evaluate Rubik’s performance with different scenarios of noisy and missing data.
3. **Scalability:** Assess the scalability of Rubik in comparison to the state-of-art methods for tensor factorization and completion.
4. **Constraints analysis:** Analyze the contribution of different constraints towards model performance.

In particular, the phenotype discovery and noise analysis are evaluated using the *Vanderbilt* data because it is real, while scalability is evaluated on both datasets. To tune hyperparameters λ_a and λ_q , we run experiments with different values and select the ones that give most meaningful results.

We compare Rubik with several baseline models as described below:

- **Marble:** This method applies sparse tensor factorization for computational phenotyping [4].
- **TF-BPP:** This is the block principle pivoting method [7] for non-negative CP tensor factorization.

⁴We assume a medication may be used to treat a specific diagnosis if both diagnosis and medication occurred within 1 week.

- **CP-APR:** This method is designed for non-negative CP Possion factorization [18].
- **WCP:** This is a gradient based tensor completion approach [8].
- **FaLRTC:** This approach recovers a tensor by minimizing the nuclear norm of unfolding matrices [5].

Phenotype Discovery

Phenotype discovery is evaluated on multiple aspects, including: 1) qualitative validation of bias tensor and interaction tensors and 2) distinctness of resulting phenotypes. We choose Marble as the competitor, since it is the only other method that generate sparse phenotypes, which is clinically important.

Meaningful Bias Tensor

A major benefit of Rubik is that it captures the characteristics of the overall population. Note that the Centers for Disease Control and Prevention (CDC) estimates that 80% of older adults suffer from at least one chronic condition and 50% have two or more chronic conditions [19]. In our bias tensor, five of the ten diagnoses are chronic conditions, which supports the CDC claim. In addition, the original data had a large percentage of patients with hypertension and related co-morbidities, such as chronic kidney disease, disorders of lipid metabolism and diabetes. Most of the elements (for diagnosis and medication modes) in the bias tensor shown in Table 2.4 are also found among the most commonly occurring elements in the original data. Based on the above observations, we can see that the elements of the bias tensor factor are meaningful and that they accurately reflect the stereotypical type of patients in the Vanderbilt dataset.

Meaningful Interaction Tensor

Next, we evaluate whether the interaction tensor can capture meaningful phenotypes. To do so, we conducted a survey with three domain experts, who did not know which model

Table 2.4: Elements of the diagnosis and medication modes in the bias tensor.

Diagnoses	Medications
Hypertension	Statins
Disorders of lipid metabolism	Loop diuretics
Heart failure	Miscellaneous analgesics
Respiratory & chest symptoms	Antihistamines
Chronic kidney disease	Vitamins
Other and unspecified anemias	Calcium channel blockers
Diabetes mellitus type 2	Beta blockers
Digestive symptoms	Salicylates
Other diseases of lung	ACE inhibitors

they were evaluating. Each expert assessed 30 phenotypes (as in Table 2.1) from Rubik and 30 phenotypes from Marble. For Rubik, we introduced four phenotypes with partial diagnosis guidance. For each phenotype, the experts assigned one of three choices: 1) YES - clinically meaningful, 2) POSS - possibly meaningful, 3) NOT - not meaningful.

We report the distribution of answers in Figure 2.1. The inter-rater agreement is 0.82, indicating a high agreement. Rubik performs significantly better than the baseline method. On average, the domain experts determined 65% (19.5 out of 30) of the Rubik phenotypes to be clinically meaningful, with another 32% of them to be possibly clinically meaningful. On the other hand, the clinicians determined only 31% of the baseline Marble derived phenotypes to be clinically meaningful, and 51% of Marble derived phenotypes to be clinically meaningful. Only 3% of Rubik derived phenotypes were determined to be not meaningful, while 18% of Marble derived phenotypes were considered not meaningful. These results collectively suggest that Rubik may be capable of discovering meaningful phenotypes.

Discovering Subphenotypes

Here we demonstrate how novel combinations of guidance information and pairwise constraints can lead to the discovery of fine-grained subphenotypes. In this analysis, we add the diagnosis guidance to *hypertension*, *type 1 diabetes*, *type 2 diabetes* and *heart failure* separately.

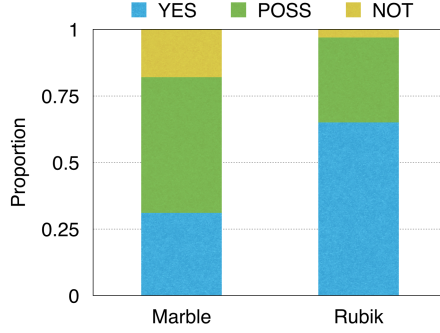


Figure 2.1: A comparison of the meaningfulness of the phenotypes discovered by Marble and Rubik.

To gain an intuitive feel for how guidance works, let us use the guidance for *hypertension* as an example. We construct the guidance matrix $\hat{\mathbf{A}}^{(2)}$ as follows. Assume the index corresponding to *hypertension* to be n . Then, we set the n^{th} entry of vector $\hat{\mathbf{A}}_1^{(2)}$ and $\hat{\mathbf{A}}_2^{(2)}$ to be one and all other entries equal to zero. The pairwise constraint will push $\mathbf{A}_1^{(2)}$ and $\mathbf{A}_2^{(2)}$ to be as orthogonal as possible (i.e., small cosine similarity). In other words, the resulting factors will share less common entries, and will thus be more distinct. In summary, by introducing multiple identical guidance factors (e.g., 2 hypertension vectors), the algorithm can derive different subphenotypes which fall into a broader phenotype described by the guidance factors.

Table 2.5 shows an example phenotype for hypertension patients that was discovered using Marble. While this phenotype may be clinically meaningful, it is possible to stratify the patients in this phenotype into more specific subgroups. Table 2.6 demonstrates the two subphenotypes discovered by Rubik with the hypertension guidance, which effectively include non-overlapping subsets of diseases and medications from the Marble derived phenotype.

In a similar fashion to the evaluation of the interaction tensor, we asked domain experts to evaluate the meaningfulness of subphenotypes. Rubik incorporated four guidance constraints (for four separate diseases), and generated two subphenotypes for each guidance constraint, resulting in eight possible subphenotypes. Domain experts were asked to eval-

Table 2.5: An example of a Marble-derived phenotype.

Diagnoses	Medications
Chronic kidney disease	Central sympatholytics
Hypertension	Angiotensin receptor blockers
Unspecified anemias	ACE inhibitors
Fluid electrolyte imbalance	Immunosuppressants
Type 2 diabetes mellitus	Loop diuretics
Other kidney disorders	Gabapentin

Table 2.6: Two subphenotypes of hypertensive patients.

A. Metabolic syndrome phenotype

Diagnoses	Medications
Hypertension	Calcineurin inhibitors
Chronic kidney disease	Insulin
Ischemic heart disease	Immunosuppressants
Disorders of lipid metabolism	ACE inhibitors
Anemia of chronic disease	Calcium channel blockers
	Antibiotics
	Statins
	Calcium
	Cox-2 inhibitors

B. Secondary hypertension phenotype

Diagnoses	Medications
Secondary hypertension	Class V antiarrhythmics
Fluid & electrolyte imbalance	Salicylates
Unspecified anemias	Antianginal agents
Hypertension	ACE inhibitors
	Calcium channel blockers
	Immunosuppressants

uate whether or not each subphenotype made sense as a subtype of the original constraint. The inter-rater agreement is 0.81. On average, the clinicians identified 62.5% (5 out of 8) of all subphenotypes as clinically meaningful, and the remaining 37.5% of subphenotypes to be possibly clinically meaningful. None of the subphenotypes were identified as not clinically meaningful. These results suggest that Rubik can be effective for discovering subphenotypes given knowledge guided constraints on the disease mode.

More Distinct Phenotypes

One important objective of phenotyping is to discover distinct phenotypes. In this experiment, we show that adding pairwise constraints guidance reduces the overlap between phenotypes. We also fix $\lambda_a = 0$ and change the weight of pairwise constraint (λ_q) to evaluate the sensitivity of Rubik to this constraint.

We first define *cosine similarities* between two vectors \mathbf{x} and \mathbf{y} as $\cos(\mathbf{x}, \mathbf{y}) = \frac{\langle \mathbf{x}, \mathbf{y} \rangle}{\|\mathbf{x}\| \|\mathbf{y}\|}$. Then we use *AvgOverlap* to measure the degree of overlapping between all phenotype pairs. This is defined as the average of the *cosine similarities* between all phenotype pairs in the diagnosis mode. The formulation for *AvgOverlap* is as follows.

$$AvgOverlap = \frac{\sum_{m=1}^R \sum_{n>m}^R \cos(\mathbf{A}_m^{(2)}, \mathbf{A}_n^{(2)})}{R(R-1)/2}$$

where $\mathbf{A}_m^{(2)}$ denotes the m^{th} column of factor matrix $\mathbf{A}^{(2)}$, which is the vector representation of the m^{th} phenotype on the diagnosis mode.

Figure 2.2 shows the change of average cosine similarity as a function of λ_q . We can see that the average similarity tends to stabilize when λ_q is larger than 10. Figure 2.2 also compares Rubik with Marble, CP-APR and TF-BPP. Note that the three competitors do not incorporate pairwise constraints and clearly lead to significantly overlap in their phenotypes. As such, we can conclude that adding the pairwise constraints can effectively lead to more distinct phenotypes.

Noise Analysis

There are multiple sources of noise in real clinical applications. For example, part of the EHR data could be missing or simply incorrect. Alternatively, the clinical guidance could be noisy. To test Rubik against such conditions, we systematically evaluate different scenarios of noise. The results are summarized over 10 independent runs.

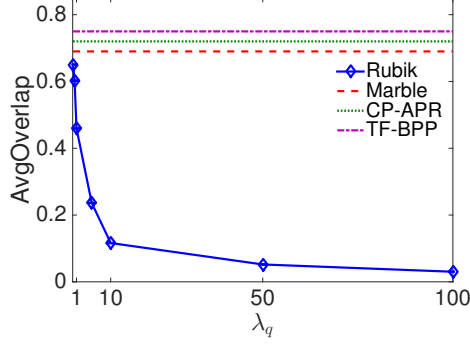


Figure 2.2: The average level of overlap in the phenotypes as a function of the pairwise constraint coefficient λ_q .

Robustness to Missing Data

The EHR data may have missing records. Consequentially, the observed binary tensor \mathcal{O} might contain missing values. To emulate the missing data, we flip the value of each non-zero cell $\mathcal{O}_{\vec{i}}$ to zero with probability p .

Let $\mathcal{T} = [\mathbf{A}^{(1)}; \dots; \mathbf{A}^{(N)}]$ be the solution without missing data and $\mathcal{T}^p = [\mathbf{A}^{p(1)}; \dots; \mathbf{A}^{p(N)}]$ be the solution with missing data level p on the input tensor \mathcal{O} . We first pair \mathcal{T}^p with \mathcal{T} using a greedy algorithm. Then we define the average cosine similarities ($AvgSim$) between \mathcal{T} and \mathcal{T}^p as

$$AvgSim(\mathcal{T}, \mathcal{T}^p) = \frac{\sum_{n=1}^N \sum_{r=1}^R \cos(\mathbf{A}_r^{(n)}, \mathbf{A}_r^{p(n)})}{NR} \quad (2.13)$$

Figure 2.3 shows the change in similarity measures with the change of missing-data level. It can be seen that Rubik performs well even if 30% of the data is missing, indicating its robustness. By contrast, the other baseline methods are not robust to missing data quantities as small as 10%.

Robustness to Noisy Data

In creating documentation in the EHR dataset, clinicians may introduce erroneous diagnosis and medication information. The implication of such errors is that the observed binary tensor \mathcal{O} will also contain noise. To emulate this setting, we randomly select zero value

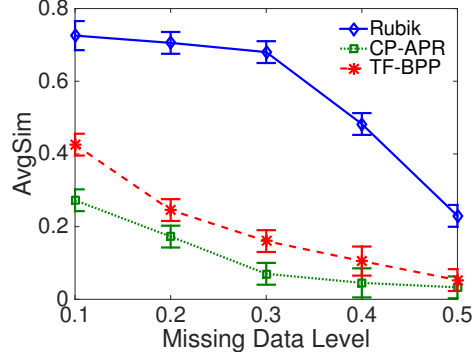


Figure 2.3: An average similarity comparison of different methods as a function of the missing data level.

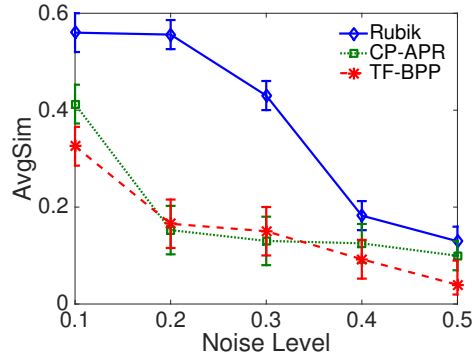


Figure 2.4: An average similarity comparison of different methods as a function of the noise level.

cells, such that the total number of selected cells is equal to the number of non-zero cells. Then, for each cell in this collection, we flip its value with probability p . The difference from introducing missing data is that we introduce multiple incorrect ones as noisy data while missing data case will remove multiple correct ones.

Figure 2.4 shows that Rubik performs well - even when the noise level is as high as 20% - 30%. One intuitive explanation for such a high tolerance is that the model is dependent on the observed tensor, as well as various constraints. In summary, Rubik is resilient to noise. As such, Rubik will provide more generalizable phenotypes than its competitors.

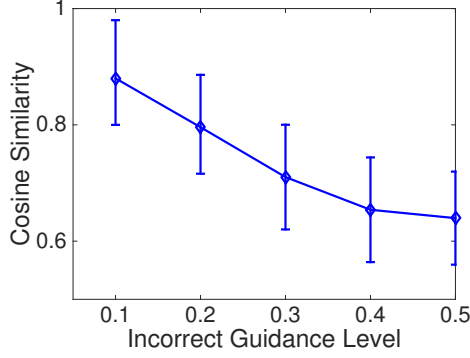


Figure 2.5: The similarity between the true solution and the solution under incorrect guidance as a function of the incorrect guidance level.

Robustness to Incorrect Guidance

At times, guidance knowledge may be incorrectly documented (or the state of medical belief may be incorrect). Take *hypertension* for example, to simulate incorrect guidance on this diagnosis, we randomly pick K (we use 4 in our experiments) entries from the corresponding column in the guidance matrix $\hat{\mathbf{A}}^{(2)}$ and set it to be one with probability p . Note K is small here for two reasons. First, in phenotyping applications we hope to achieve sparse solutions, which implies that the guidance should also be sparse. Second, medical experts do not typically have much guidance on a particular phenotype.

We then compare the *cosine similarity* between the phenotypes obtained by correct and incorrect guidance. Figure 2.5 demonstrates that as the level of incorrect guidance increases the cosine similarity slowly decreases. Therefore, even when a significant portion of the guidance is incorrect, the phenotype can remain fairly close to the original.

Parameter Tuning

Rubik computes the sparse factor representation using a set of predefined thresholds γ_n , which provide a tunable knob to adjust the sparsity of the candidate phenotypes. In our experiment, we numerically evaluate the sensitivity of the sparse solution with respect to different threshold values. To do so, we randomly downsample the tensor \mathcal{O} by 50% and

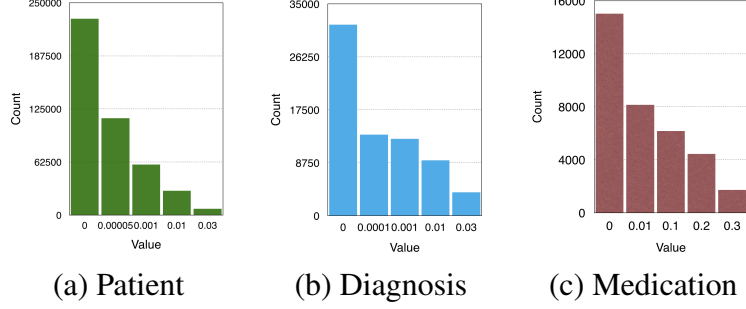


Figure 2.6: The count of non-zero elements along the three modes as a function of the thresholding parameters.

run Rubik on this smaller tensor. The above procedure is averaged over 10 independent runs.

Figure 2.6 shows the distribution of the non-zero factor values along the three modes. For all three plots, there is a noticeable difference in size between the first two bins, which suggests the threshold occur at the start of the second bin. Thus, for the paper, we set $\gamma = 0.00005, 0.0001$ and 0.01 for the patient, diagnosis and medication modes, respectively.

Scalability

We test the scalability of Rubik on both datasets. We randomly sample a different number of patients from the datasets and construct the tensors. The results are summarized over 10 independent runs and the performance is measured in runtime (seconds).

Figure 2.7 reports the runtime comparison of different tensor factorization methods on the Vanderbilt dataset. We can clearly see the advantage of our framework over CP-APR. Specifically, the runtime is reduced by 70%. TF-BPP is comparable to Rubik on the Vanderbilt dataset. For the CMS dataset, Rubik is around six times faster than the two baselines.

For the tensor completion task, Figure 2.7 shows that Rubik is nearly 5-7 times faster than WCP and FaLRTC. Figure 2.8 further demonstrates Rubik’s superiority over these methods. Note that FaLRTC and WCP will reach their maximum default number of iterations before the algorithm actually converges on the CMS dataset, so that more time is

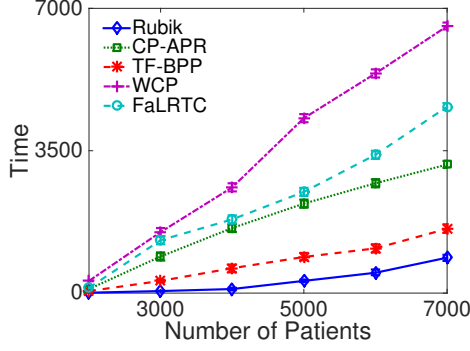


Figure 2.7: A runtime comparison of different methods on the Vanderbilt dataset as a function of the number of patients.

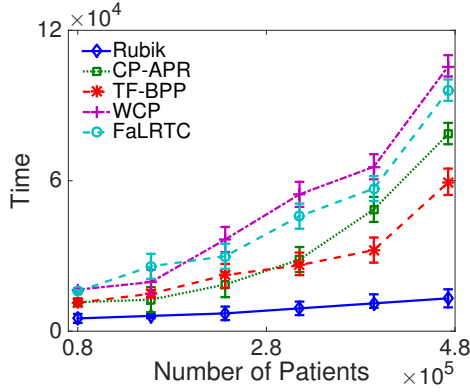


Figure 2.8: A runtime comparison of different methods on the CMS dataset as a function of the number of patients.

actually needed to complete the baselines.

Constraints Analysis

We investigate the sensitivity of Rubik with regard to three constraints: completion, guidance and pairwise constraints. We begin by running Rubik under baseline conditions without any constraints. Then, we iteratively add constraints independently and compute *AvgSim* (Eq. 2.13) between the solution obtained with that constraint and without that constraint. Then, we normalize all *AvgSim* scores and let them sum to one. The contribution of each constraint is measured by the normalized *AvgSim* score. Note that we did not incorporate guidance information into Rubik on the CMS dataset.

Figure 2.9 reports the proportion of each constraint’s contribution to the overall model.

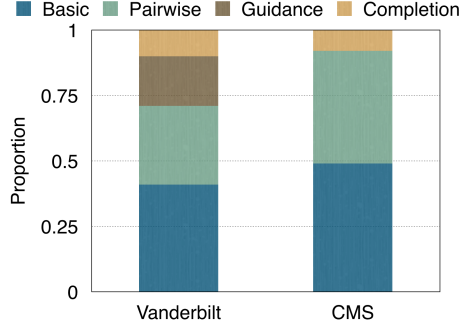


Figure 2.9: Proportion of contribution of each constraint.

The bars labeled *Basic* represent the baseline performance of the model without any constraints imposed. The bars labeled *Pairwise*, *Guidance* and *Completion* represent the contributions of the corresponding constraints, respectively. We see that all of the constraints represent a significant contribution to the model’s overall performance. Amongst the constraints, the *pairwise constraint* provides the largest contribution to the performance.

2.1.2 Related Work

Non-negative Tensor Factorization. Kolda [20] provided a comprehensive overview of tensor factorization models [20]. It is desirable to impose a non-negativity constraint on tensor factorizations in order to facilitate easier interpretation when analyzing non-negative data. Existing non-negative matrix factorization algorithms can be extended to non-negative tensor factorization. Welling and Webber proposed multiplicative update algorithm [21]. Chi and Kolda [18] proposed nonnegative CP alternation Poisson regression (CP-APR) model. Kim et al. [7] proposed an alternating non-negative least square method with a block pivoting technique.

Constrained Factorization. Incorporating guidance in tensor factorization has drawn some attention over the past few years. Carroll et al. [22] used linear constraints. Davidson [9] proposed a framework to incorporate pattern constraints for network analysis of fMRI data. However, this method is domain specific and might not be applicable to other areas such as computational phenotyping. Narita [23] provided a framework to incorporate

auxiliary information to improve the quality of factorization. However, this work fails to incorporate nonnegativity as constraints to the factor matrix. Solving for non-negativity constraints usually requires a nontrivial calculation step.

Coupled Factorization. In this case, we have additional data matrices that share the same dimension as the tensor. The goal is to jointly factorizing the tensor and matrices [24]. Acar et al. [6] used first order optimization techniques. There are also scalable algorithms on Hadoop [25, 26]. However, this framework does not directly cover all the constraints we need in our application.

Tensor Completion. Liu et al. [5] and Signoretto [27] generalized matrix completion to the tensor case to recover a low-rank tensor. They defined the nuclear norm of a tensor as a convex combination of nuclear norms of its unfolding matrices. Tomioka and Suzuki [28] proposed a latent norm regularized approach. Liu et al. [29] substituted the nuclear norm of unfolding matrices by the nuclear norm of each factor matrix of its CP decomposition. A number of other alternatives have also been discussed in [30, 31, 32]. However, these methods suffer from high computational cost of SVDs at each iteration, preventing it from scaling to large scale problems.

Without minimizing the nuclear norm, Acar et al. [8] proposed to apply tensor factorization in missing data to achieve low rank tensor completion. However, none of these methods are guaranteed to output a non-negative factor matrix for each mode or non-negative tensor. As a consequence they are not applicable to our non-negative tensor setting. Incorporating non-negativity as constraints to factor matrices, Xu et al. [33] proposed an alternating proximal gradient method for non-negative tensor completion. However, they did not take the guidance information into account and their gradient based method is not scalable to large datasets.

In summary, existing tensor factorization and completion methods are not applicable to computational phenotyping.

2.1.3 Conclusion

This paper presents Rubik, a novel knowledge-guided tensor factorization and completion framework to fit EHR data. The resulting phenotypes are concise, distinct, and interpretable. One distinguishing aspect of Rubik is that it is able to discover subphenotypes. Furthermore, Rubik captures the baseline characteristics of the overall population via an augmented bias tensor.

In the experiments, we demonstrate the effectiveness of adding the guidance on diagnosis. Rubik can also incorporate guidance from other sources such as medications and patients. We also demonstrate the scalability of Rubik on simulated EHRs in a dataset with millions of records. Rubik can potentially be used to rapidly characterize and manage a large number of diseases, thereby promising a novel solution that can benefit very large segments of the population. Future work will focus on evaluating Rubik on larger datasets and conduct larger medical validations with experts.

2.2 Applications of phenotype discovery algorithms

We propose the application of phenotyping towards the extraction of heart failure subtypes from EHR data.

2.2.1 Segmentation of heart failure patients via nonnegative tensor factorization

Abstract

Background: Heart failure is heterogenous in nature and often presents with multiple comorbidities, making it difficult for healthcare providers to determine targeted treatment plans for specific patients.

Objective: To extract phenotypic subgroups from a cohort of congestive heart failure patients and to demonstrate the extracted subgroups represented significantly distinct risk profiles.

Methods: An unsupervised machine learning method, non-negative tensor factorization, was used to discover phenotypic subgroups from electronic health record (EHR) data for diagnoses and medications and labs. Patient data from EHR time series was represented in a 3-way tensor capturing co-occurrences of event types. An observational dataset of 4,370 patients with heart failure diagnosis was extracted from the electronic health record system of Sutter Health Palo Alto Medical Foundation, an ambulatory care practice, and used for this study. Each patient has an average follow-up time of 2 years after the heart failure diagnosis date and 4 years before that date. Phenotypes were computed for all patients, as well as separately on patients with heart failure with reduced ejection fraction (HFrEF) and heart failure with preserved ejection fraction (HFpEF).

Results: Five phenotypes were discovered from each of the three subsets of patients: all HF patients, patients with HFpEF, and patients with HFrEF. There were major differences in the phenotypes of HFpEF as compared with HFrEF. Two of the phenotypes of HFpEF exhibited features that were not present in any of the phenotypes of HFrEF. Specifically, there was a phenotype of HFpEF predominantly characterized by patients with diabetes and associated complications, as well as a phenotype of HFpEF predominantly characterized by patients with musculoskeletal problems. Further analysis of the phenotypes with respect to lab values indicate different characteristics which may lead to different trajectories.

Conclusions: Via a non-negative tensor factorization based phenotyping algorithm, cardiovascular patients were automatically segmented and characterized.

Introduction

Phenotyping may be particularly important for pre-diagnostic HFpEF, as it is considered to be etiologically more heterogeneous than HFrEF[34]. To date, no study has attempted to classify pre-diagnostic HF into phenotypes using comprehensive EHR data. We propose methods that allow patients to be defined by more than one phenotype as it is likely that causal processes are not independent. We propose to build on tensor factorization methods

for phenotyping that we have pilot tested on a Geisinger EHR data in identifying HF patient phenotypes.

While there are several studies performing stratification of patients, most of them perform stratification based on their perceived disease severity derived from co-morbidities[5, 35]. These assumptions may be based on problems or lab values present in a patient. Recently, Luo et al. suggested the usage of tensor factorization for unsupervised subtyping of patients with heart failure with preserved ejection fraction[36]. Given the vast amount of lab values, symptoms and other characteristics of the patient (characteristics inherent to the patient, rather than disease labels), we believe that nonnegative tensor factorization provides an effective unsupervised computational method to segment or stratify the broader cardiovascular disease population into meaningful subgroups that would be likely to behave similarly to each other[37, 12, 4].

Patients segmented in such a manner would aid physicians in developing targeted treatment regimens and ultimately lead to better outcomes. Furthermore, such an unsupervised method would help researchers to discover novel phenotypes that are difficult to uncover with a manual approach. This is crucial as there are likely many phenotypes of cardiovascular patients who are heterogeneous but whose disease is not very well characterized (e.g. patients with heart failure with preserved ejection fraction). (Wilcox and Yancy 2016)

Methods

Data

In this subsection we describe the nature of the data and the processing required for the phenotype discovery.

Population and Source of Data

The source population for this study was primary care patients from Sutter Palo Alto Medical Foundation (PAMF) Clinics, multispecialty group practices with large primary care

practices. Sutter PAMF includes 497 primary care providers in the northern California Bay Area and coastal areas that provide care to 700,000 patients.

EpicCare EHR was installed at Sutter-PAMF in 1999. Assignment of patients to a PCP is documented in an EHR structured field. Incident HF cases were identified from Sutter-PAMF between January 1, 2001 and December 31, 2010 and from Sutter PAMF from May 16, 2000 to May 23, 2013.

Cohort Construction

Definition of Cases

Criteria for incident onset of HF were adopted from [38] while criteria for HF subtypes were defined in detail elsewhere [38, 39, 36], but relied on qualifying ICD-9 codes for HF with a minimum of three clinical encounters occurring within 12 months of each other. Qualifying cases must be assigned to a PCP in the appropriate EHR structured field. Qualifying ICD-9 codes included 398.91, 402.01, 402.11, 402.91, 404.01, 404.03, 404.11, 404.13, 404.91, 404.93, 428.0, 428.1, 428.20, 428.21, 428.22, 428.23, 428.30, 428.31, 428.32, 428.33, 428.40, 428.41, 428.42, 428.43, 428.9, EP427, EP428, EP429, EP431, EP432, and EP433. The date of diagnosis (HFDx) was assigned as follows. If the time between the first and second HF diagnosis codes is less than or equal to 365 days, and the time between the first and third HF diagnosis codes was less than or equal to 547 days, then select the first HF diagnosis code as the HFDx. Otherwise, select the second HF diagnosis code as the HF diagnosis date (HFDx). Analysis was limited to patients who were 40 to 84 at the time of HF diagnosis. A total of 4,370 incident HF cases were identified from Sutter-PAMF over the time period from 2000 to 2013. In each distinct record, the ejection fraction status was determined from text mining via natural language processing of clinical notes that document physician interpretations of echocardiogram recordings in the electronic health record happening. Specifically, natural language processing algorithms

were used to assign a label of either Preserved or Reduced to indicate ejection fraction status from each separate record occurring within 6 months (before and after) HFDx (Figure 2.10).

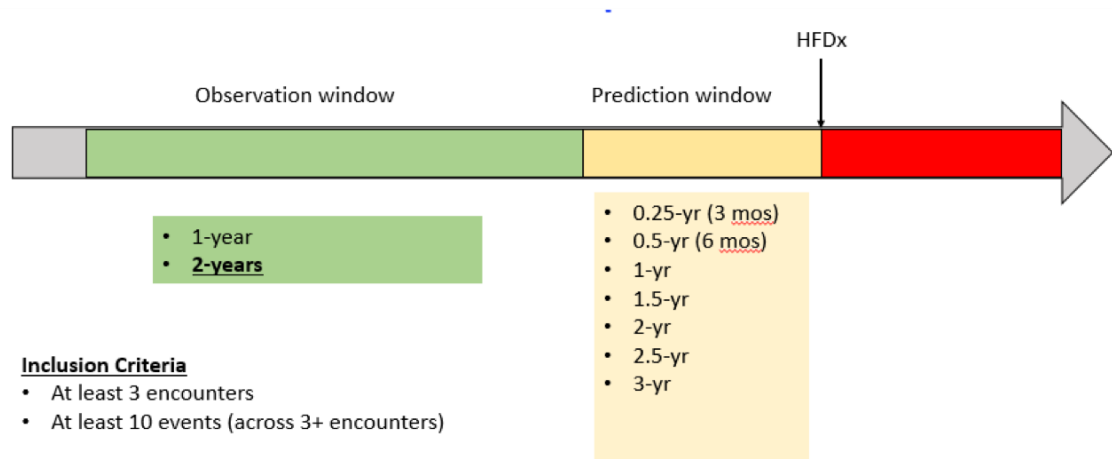


Figure 2.10: Criteria for inclusion of data.

Definition of HFrEF, HFpEF and HF with uncertain EF

Class labels are defined for three separate subsets of the HF patients: heart failure with preserved ejection fraction (HFpEF), heart failure with reduced ejection fraction (HFrEF) and heart failure with uncertain ejection fraction. There are total of 4370 case patients with congestive heart failure. Of the case patients, 3553 have an ejection fraction record. Of these 3553 patients, 2449 have ejection fraction status within a one year period of time spanning 6 months before or after the heart failure diagnosis date. That is, all ejection fraction status recordings from the time period HFDx-183 days to HFDx+183 days are extracted and used to determine reduced EF and preserved EF status. A patients HF status is labeled as HFpEF if all ejection fraction records have the value Preserved, while a patients HF status is labeled as HFrEF if all ejection fraction records have the value Reduced. A patients HF status is labeled as HF with uncertain EF if there are values of Preserved and Reduced among all of that patients records. Of the 2449 case patients with ejection fraction

recordings, there are 1110 patients where all events are reduced, 1019 patients where all events are preserved, and 320 patients with both reduced and preserved events

Note that 78 patients were removed from the study because a class label of Reduced or Preserved cannot be reasonably inferred. This leaves us with 1117 CHF cases with preserved ejection fraction and 1254 CHF cases with reduced ejection fraction

Feature grouping

Features for diagnosis codes were grouped according to the Hospital Cost and Utilization Project Clinical Classification Software (CCS) grouping.

Filtering features

We excluded diagnoses that were not directly indicative of a medical state or condition. We did not exclude any medications or lab values from the analysis.

Filtering patients

All patients with less than 5 encounters within the observation window were excluded.

Experiment setup

Three separate arrays of phenotyping were performed. Phenotypes were extracted from the following subgroups of patients: all CHF patients, patients with HFrEF, and patients with HFpEF (Figure 2.11).

Phenotyping via non-negative tensor factorization

Non-negative tensor factorization is used to decompose the tensor representation of patient data into sub-parts (figure 2.12).

Results

Figure 2.13 shows the 5 phenotypes derived from patients with HFpEF, and Figure 2.14 shows the 5 phenotypes derived from patients with HFrEF. There were major differences in the phenotypes of HFpEF as compared with HFrEF. Two of the phenotypes of HFpEF exhibited features that were not present in any of the phenotypes of HFrEF. Specifically,

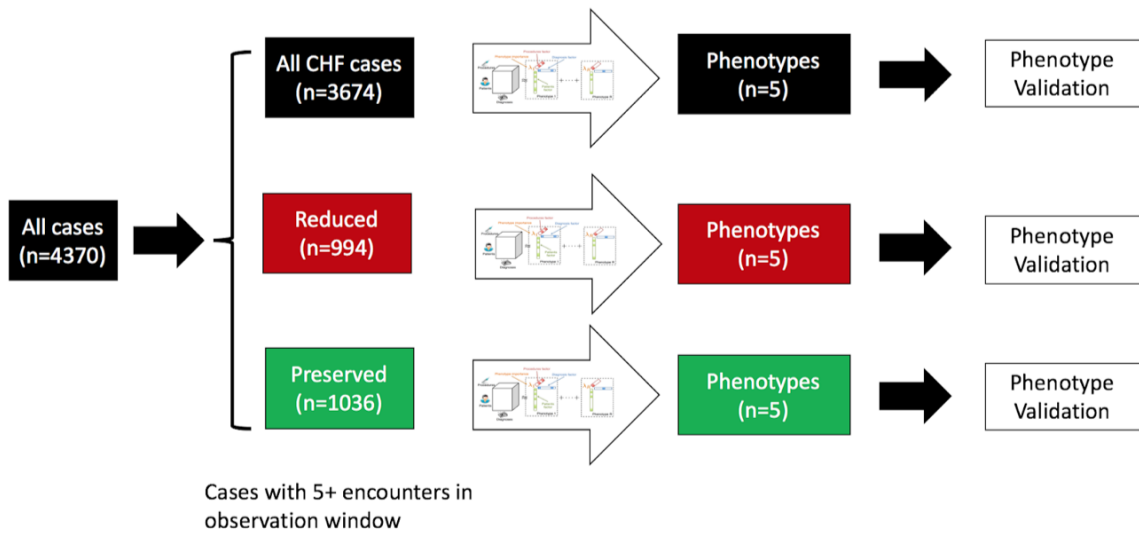
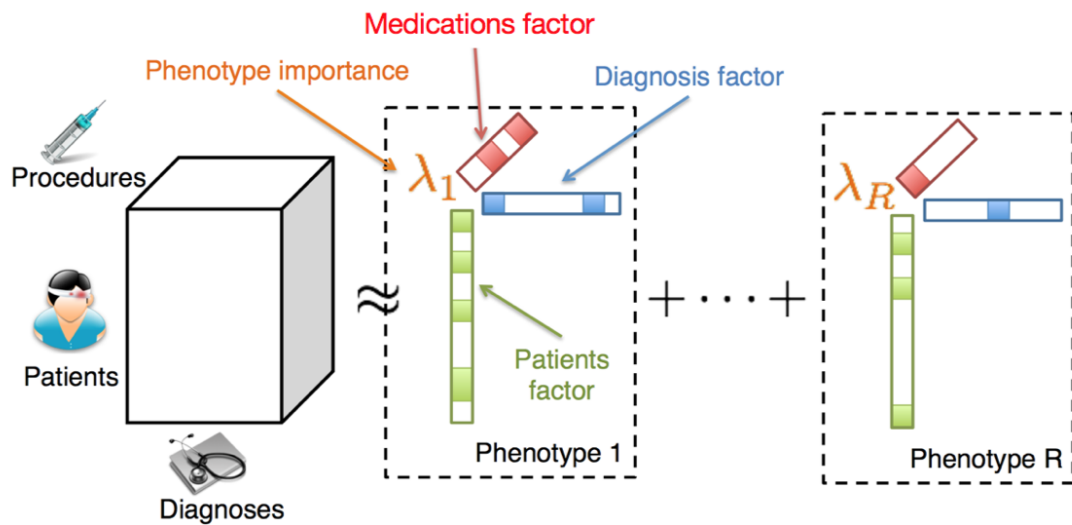


Figure 2.11: A graphical representation of the experimental setup.



- Non-zero elements map to clinical characteristics of that phenotype

Figure 2.12: Graphical representation of non-negative tensor factorization.

there was a phenotype of HFpEF predominantly characterized by patients with diabetes and associated complications, as well as a phenotype of HFpEF predominantly characterized by patients with musculoskeletal problems. Further analysis of the phenotypes with respect to lab values indicate different characteristics which may lead to different trajectories.

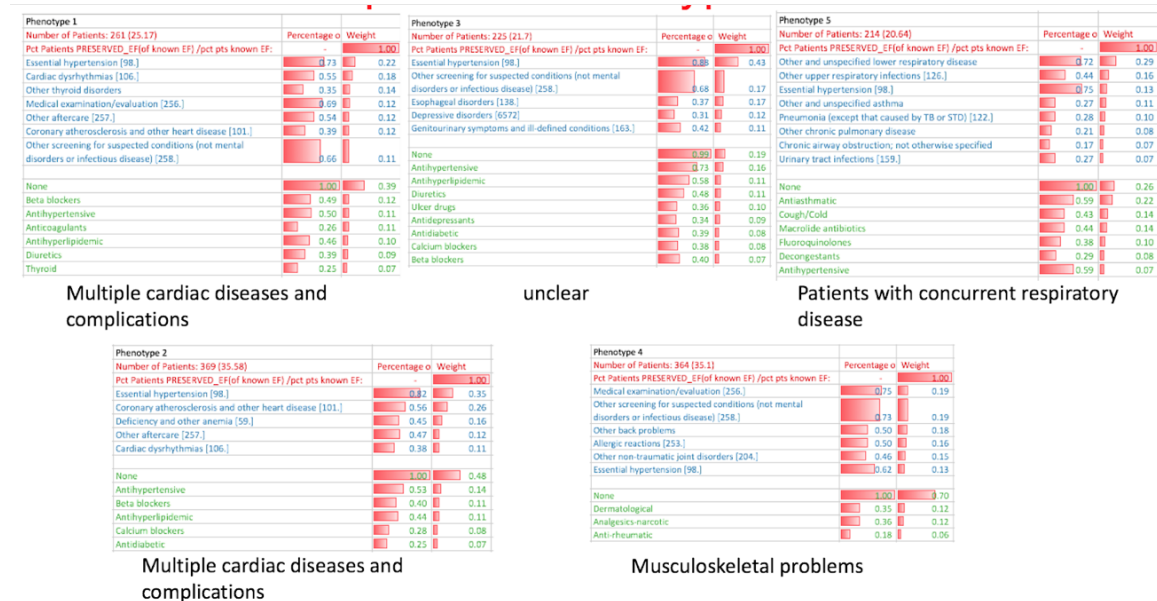


Figure 2.13: Phenotypes of patients with heart failure with reduced ejection fraction (HFrEF).

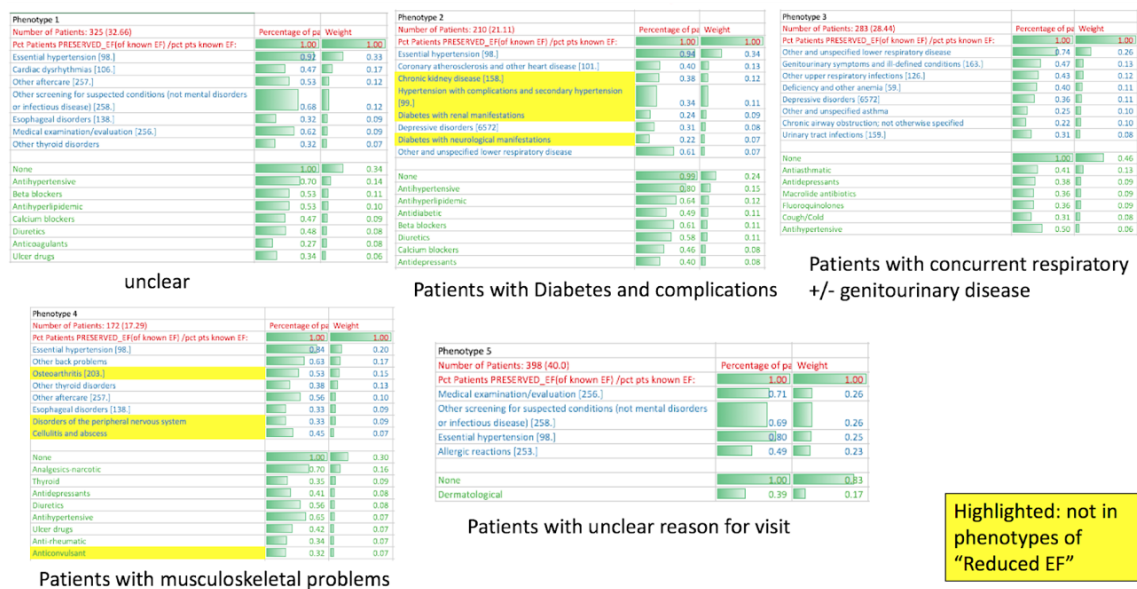


Figure 2.14: Phenotypes of patients with heart failure with preserved ejection fraction (HF-pEF).

CHAPTER 3

PREDICTIVE MODELING

3.1 Applications

In the area of predictive modeling, we propose the novel usage of deep learning for early detection of heart failure. To this end, we assessed the efficacy of recurrent neural networks for early detection of heart failure from periods of 3 months to 2 years before heart failure diagnosis.

3.1.1 Recurrent neural networks for early detection of heart failure from longitudinal EHR data: implications for temporal modeling with respect to time before diagnosis, data density, data quantity and data type.

Abstract

Background: Using electronic health records (EHR) data to predict events and onset of diseases is increasingly common. Previous efforts have to study tradeoffs between data requirements and model utility have typically been limited to traditional machine learning models such as logistic regression and random forest.

Methods and Results: We assessed the relative performance of recurrent neural network (RNN) and traditional machine learning models trained to detect pre-diagnostic heart failure (HF) in primary care patients using longitudinal EHR data. Model performance was assessed in relation to data requirements defined by: the combination of different data domains (data diversity), the number of patient records in the training data set (data quantity), the number of patient encounters (data density), the prediction window length (time before HF clinical diagnosis), and the observation window length (duration of observation prior to prediction window). A total of 4,370 incident HF cases and 30,132 gender,

age-category, and clinic matched controls were used for modeling. Compared to traditional machine learning models, RNN model is supreme when data is sparse (less diversity) given the same training dataset. In general, RNN model performance improved as: 1) the data diversity increases, where the demographics, vital signs, diagnoses medications and social history were the most important data domains; 2) data quantity increases. In our case, minimum 30% of the patients when data are sparse/thin (single domain), and using more than 80% of the patients when all data domains are used; 3) data were confined to patients who had 5 or 10 or more phone or face-to-face encounters given fixed 2 years observational window; 4) prediction window size decreases, especially when less than 2 years; and 5) observation window size increases, especially after 1 year.

Additionally, our study sheds light on optimal training set sizes needed for RNN models for heart failure prediction. Namely, in conditions where data are sparse or thin, a much smaller training set size is required in order to achieve optimal AUC compared to the condition where data are more diverse and include more features.

Conclusions: These empirical findings suggest that recurrent neural networks are effective for heart failure prediction and suggest possible guidelines for the minimum amount and type of data needed to train effective disease onset predictive models using longitudinal EHR data.

Introduction

Heart failure is a highly prevalent and costly disease that is strongly associated with mortality. It is estimated that 5.7 million Americans have heart failure, which leads to a \$31 billion yearly cost. Furthermore, across all affected patients, heart failure bears a 50% death rate within 5 years. There is an increasing amount of prior evidence that having certain physiological factors or conditions are associated with increased mortality risk. For example, increases in blood pressure as small as 2 mm Hg (either systolic or diastolic) have been linked to increased rates of adverse clinical outcomes such as renal failure at the

population level (James et al. 2014; Chobanian et al. 2003). As heart failure is a complex heterogeneous disease with unconfirmed pathophysiology and characterization (Shah, Katz, and Deo 2014), attempts have been made to stratify patients with unsupervised learning via Gaussian mixture models (Shah et al. 2015), and with various rule-based approaches based on differences in heart structure [40]. Previously, a study conducted by Ng et al. [41, 42, 43] studied the effect of various predictive modeling parameters on prediction performance, represented by the area under the receiver operating curve (AUC). Ng et al. tested prediction window length, observation window length, data diversity, training data size and data density in two traditional machine learning models: logistic regression and random forest. The main limitation of Ng et al.'s approach was that it did not incorporate temporal information, as traditional machine learning methods such as logistic regression and random forest represent patient events as aggregated counts of event occurrences. However, capturing temporal information is valuable for early detection of disease. Recurrent neural network is a temporal model that encodes time-stamped events and learns latent representations that can be leveraged for classification task. Previously, Choi et al. implemented a recurrent neural network approach (gated recurrent unit) for classification and early detection of heart failure from longitudinal electronic health record data [44], an interpretable RNN-based model leveraging attention mechanism. However to our knowledge, there is no literature to evaluate RNN-based model compared to the traditional machine learning models in prediction of heart failure, and assess to what extent the RNN-based models outperform traditional models given the same modeling parameters. In this study, we seek to extend Ng et al.'s work to include temporal RNN models and to assess specific sensitivities of modelling parameters. We compare performance of RNN models (GRU) to traditional machine learning models with respect to variation of model parameters, data density, data quantity, data domain, training size, observation window length. Furthermore, we examine the tradeoffs among those parameters, particularly data domain and training size, and assess the optimal combination of those two parameters. In particular, we find that RNN

models add tremendous value under conditions where the data are sparse, and we provide valuable insight on the optimal training size that allows RNN to perform well given a single data domain or combination of multiple data domains.

Methods

We examined the performance of recurrent neural network models in comparison with traditional machine learning models, logistic regression and random forest, trained for the early detection of heart failure in longitudinal electronic health record data from primary care clinics in a health care system. Under all predictive modeling algorithms, the predictive performance was evaluated by the area under the curve (AUC).

Study design, Population and Source of Data

A nested case-control design is applied in the primary care population from Sutter Palo Alto Medical Foundation (PAMF) Clinics from which longitudinal EHR data was accessed. PAMF has multispecialty group practices, including 497 primary care providers in northern California Bay Area and coastal areas that provide care to 700,000 patients. EpicCare EHR was installed at Sutter-PAMF in 1999. Assignment of patients to a PCP is documented in an EHR structured field. Incidence cases and controls were identified from Sutter-PAMF between January 1, 2001 and December 31, 2010.

Cohort Construction

Definition of Cases

Criteria for incident onset of HF were adopted from [Gurwitz et al., 2013] [38] while criteria for HF subtypes were defined in detail elsewhere [38, 45, 46], but relied on qualifying ICD-9 codes for HF with a minimum of three clinical encounters occurring within 12 months of each other. Qualifying cases must be assigned to a PCP in the appropriate EHR structured field. Qualifying ICD-9 codes included 398.91, 402.01, 402.11, 402.91,

404.01, 404.03, 404.11, 404.13, 404.91, 404.93, 428.0, 428.1, 428.20, 428.21, 428.22, 428.23, 428.30, 428.31, 428.32, 428.33, 428.40, 428.41, 428.42, 428.43, 428.9, EP427, EP428, EP429, EP431, EP432, and EP433. The date of diagnosis (HFDx) was assigned as follows. If the time between the first and second HF diagnosis codes is less than or equal to 365 days, and the time between the first and third HF diagnosis codes was less than or equal to 547 days, then select the first HF diagnosis date as the HFDx. Otherwise, select the second HF diagnosis code as the HF diagnosis date (HFDx). Analysis was limited to patients who were 40 to 84 at the time of HF diagnosis. A total of 4,370 incident HF cases were identified from Sutter-PAMF over the time period from 2000 to 2013.

Selection of controls

Controls are selected from same clinics in PAMF. We identified all controls using follow criteria: a) Controls first PCP visit is within 365 days of the cases first PCP visit (i.e., to control for secular factors in diagnosis);

b) Control has at least one office encounter within 90 days of the cases HFDX date (i.e., control was an active primary care patient at the same time as the case);

c) Control does not have a HF diagnosis on or before the HFDX date of the case or within a time window (182 days) after the HFDX date;

d) Control is the same sex/gender as the case;

e) Controls age (at first PCP visit) is within +/- 5 years of the cases age

f) The time between the controls first PCP visit and the case HFDX date ≤ 547 days (to ensure there is enough data for modeling).

We identified all potential controls for each case, and randomly select a maximum of 10 controls for case. We are not able to identify matched controls for 172 cases, which are excluded from the study. About 1% (n=42) of cases had 1 to 9 matched controls, and for the rest 4201 cases we are able to identify at least 10 controls. The above process resulted in a total of 30,132 controls.

Data Extraction and Grouping

Longitudinal EHR encounter data were extracted for all cases and controls, comprising five data domains: demographic, diagnosis, medication, social history, and vitals (Table 1). Encounters included outpatient office visits and phone visits. For the diagnosis data domain, ICD-9 codes from outpatient office visit or phone visits were grouped by Clinical Classifications Softwares (CCS), developed by Agency for Healthcare Research and Quality (AHRQ). We adopted CCS level 3 from multiple-level CCS model to group 5379 unique ICD-9 codes into 363 unique groups. For the medications domain, 7186 normalized drug names (i.e. combining all branded names and the generic name for a medication) were grouped into 93 unique therapeutic subclasses using the Anatomical Therapeutic Chemical Classification System[1]. Dose information was not used. For demographics domain, sex, race, Hispanic origin, marital status were categorized first (Table 1), and then convert each category to a boolean variable. Social history domain mainly included smoking and tobacco type (i.e. cigarette, pipes, cigars, chewable, etc.), alcohol intake status (yes, no, not asked), illicit drug use (snuff, IV drug user), sexual history (sexually activity status, birth control usage) and other features (use of implants, injection, inserts, rhythm, sponge, surgical). Numerical value of vital domains were categorized using clinical validated thresholds (i.e. systolic BP is categorized into a binary variable, high if the value is over 140 mmHg, body mass index is categorized into a binary variable, high if the value is over 30). The details of categorization of all features are described in Table 1.

Feature Construction and Missing data

A feature vector for a patient was derived from the EHR data attributed by the patient in the observation window. In this study, events from the same feature, (e.g. multiple diagnosis of diabetes on different dates) were aggregated to a boolean variable (1/0) and utilized in logistic regression and random forest model, if no associated measure (i.e. such as no ICD-9 codes associated with diabetes for the patient) in the observation window the feature is set to 0. Features used in RNN model, however, considered temporal sequence of

the same feature on different dates, a vector of this feature was constructed based on each event. For example, diabetes diagnosis appeared in the first 4 times in 5 total encounters for a patient, then for this patient, the diabetes diagnosis feature is (1, 1, 1, 1, 0). Table 1 shows the number of grouped features used in the model. For continuous measures in vitals domain (i.e. systolic BP, diastolic BP, BMI), we took the mean of the vital measures within observational window, and then categorized it. There were no missing in vitals. There were no missing for age, sex, race, ethnicity data, for marital status, if missing, then created a separate category denoted as missing, and converted this category to a boolean variable. The same rules were applied to social behavior variables.

Predictive Modeling

For L1-regularized logistic regression and random forest models, the information gain measure is used to select the top 200 most discriminating features from the 570 grouped features (Bishop 2006). Interactions between the variables were not considered. Internal cross validation was performed to determine the final hyper-parameters: the number of features in the logistic regression model and number of trees in the random forest model (Table 1). Ten-fold cross-validation was used to train the predictive model and to obtain the measure how well the models perform in the independent data sets. The process was repeated 20 times to capture variation of fold splits. The performance of predictive models were assessed by mean of the receiver operating curve (AUC) from those repeated cross-validation folds. For RNN, the data set was randomly split into 75% training set and 25% testing set, maintaining case to control ratio in training set and testing set. Five bootstrap iterations was used to make efficient use of data. To measure prediction performance, the area under the receiver operating curve (AUC) is computed on the predictions made in the testing sets. We conducted a sequence of experiments to examine following data parameters in machine learning models (RNN-GRU, logistic regression, and random forest) and assessed their effect on prediction performance. For all experiments, the prediction window length

was fixed to 1 year and the observation window length was fixed to 2 years,

Data domain: Five different domains of data (diagnosis, medications, social history, demographic, and vitals) were individually assessed, followed by combination of the data domains, and ultimately all five data domains were used in the models to evaluate their performance. For this specific experiment, we used the patients with at least 5 encounters in the observation window.

Data density: varied with respect to inclusion of patients with at least one encounter or above to 70 or more in the observation window. All data domains were used in this experiment. For LR and RF models, 5 iterations of 10-fold cross validation for the model were performed and mean AUC was compared. For RNN-GRU model, 5 iterations on a 75% training and 25% testing data split was performed.

Training set size (data quantity) and data domain: varied in the amount of randomly selected training data that ranges from 100 to all patients (34,502), and evaluated on testing data set while keeping the case/control ratio constant. This experiment was performed in selected individual data domain, varying from a domain with simplest features to the most complicated data domain such as: 1) vital signs only (15 features), 2) medications only (93 grouped features for LR and RF, and 7189 features for RNN). We compared the model performance of using single domain with all data domains. For the experiments with all domains, 5 iterations of each model was performed and AUC was compared amongst them. For the experiment with vital signs and medications using RNN model, 20 iterations of the RNN were run on a 75% training and 25% testing data split.

See Figure ?? for graphical depiction of the process.

Predictive model algorithm descriptions

The formulations for all predictive modeling strategies are described below.

Gated recurrent unit (RNN-GRU)

A recurrent neural network (RNN) based model called gated recurrent unit (GRU), is used

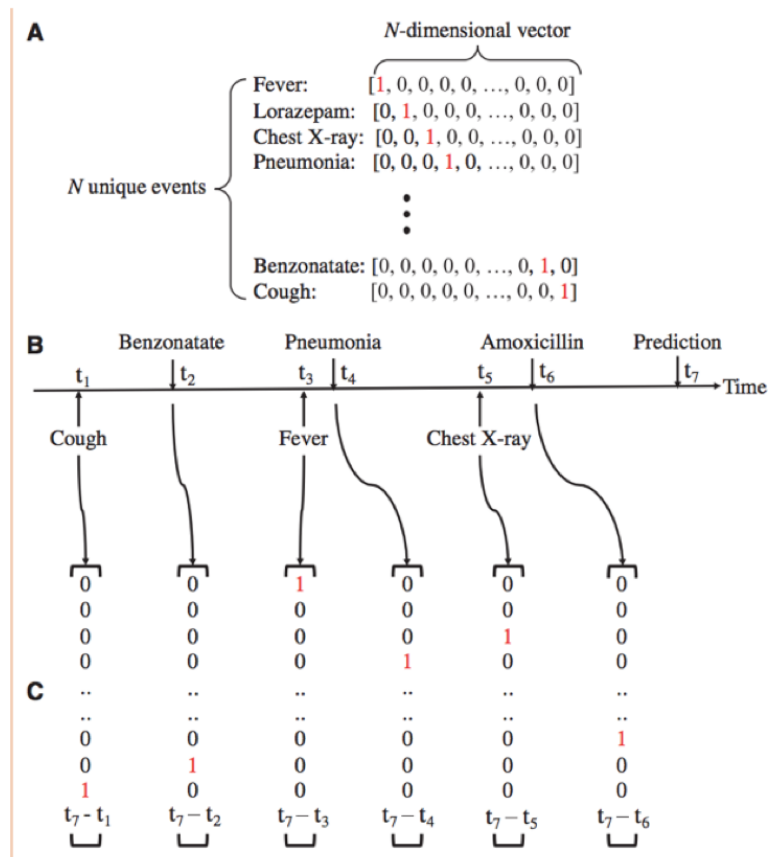


Figure 3.1: One-hot encoding of features included in model. Note: in our implementation, all 5 data domains are concatenated into single vector.

to model patient events as one-hot encoded vectors (Figure 3.1). For any given patient, given a sequence of clinical encounters of length T , the GRU model accepts as input a vector x_t , at each time step t . Information is stored at each time point $t = 0, \dots, T$, in a hidden layer with predetermined size. The state of the hidden layer changes over time from 0 to T . After all events are seen, a logistic regression computes a scalar value y , from the output of the hidden layer at the last time step T . The value of y is transformed into a class label (1 - indicating a heart failure prediction, 0 - indicating a prediction without heart failure). Figure 3.2 depicts this process.

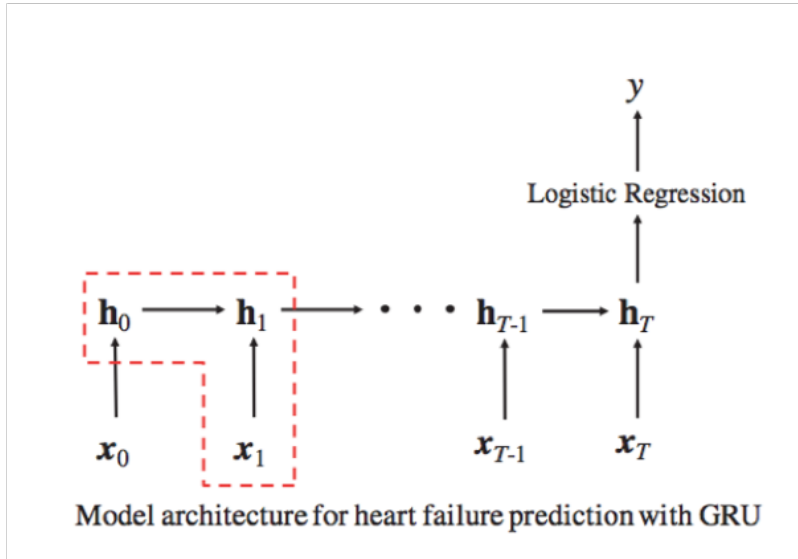


Figure 3.2: Graphical depiction of model architecture for heart failure prediction with GRU.

Logistic Regression

In the experiments, we used L1-regularized logistic regression [47], a popular parametric machine learning method with the most interpretability. The implementation from the scikit-learn framework was utilized [48]

Random Forest

Random forest model represents a computational efficient and robust approach that naturally combines bagging and feature selection in the algorithm [49]. Internal cross-validation was used to determine the number of trees used in the model. The implementation from

the scikit-learn framework was utilized [48].

All algorithms were implemented using Python.

Results

Performance as a function of data domain

Figure 3.3 shows the performance of the prediction models using grouped features in each single data domain, and combination of data domains. In single data domain, diagnosis has the best performance, followed by vitals and demographics. More specifically, diagnosis data domain that composes 363 grouped features have significant better performance across all models (AUCs \geq 0.71), especially for RNN model (AUC=0.758), which is superior to other methods and in other single data domain. For traditional model without considering temporal information, AUC for LR model is 0.743. On the other hand, social history data domain (i.e. 55 initial features) has lowest prediction power across all prediction models (AUCs \leq 0.65), and there is no significant difference between models considering temporal (RNN models, AUC=0.643) and those not (LR model, AUC=0.641). Interestingly, in the medication data domain that has 93 grouped features, prediction performance varies the most significantly among models: AUCs for the model that deals temporal information (i.e. RNN, AUC of 0.689) is much higher than those traditional models (LR or RF, AUC=0.581, 0.617 respectively) that no temporal information is considered.

As assessing the performance in combination of data domains starting with combination of demographic and vitals domains (e.g. 59 features), the best performance is found in RNN-GRU model (AUC=0.701), and AUC for LR model is 0.697. By adding diagnosis data domain (e.g. 422 features), the model performance for all models improved significantly (AUCs \geq 0.77), however, model performance does not vary significantly for RNN models versus traditional models (AUC for RNN-GRU=0.792, AUC for LR = 0.783). Further adding additional data domains (medication domain first, then social history domain),

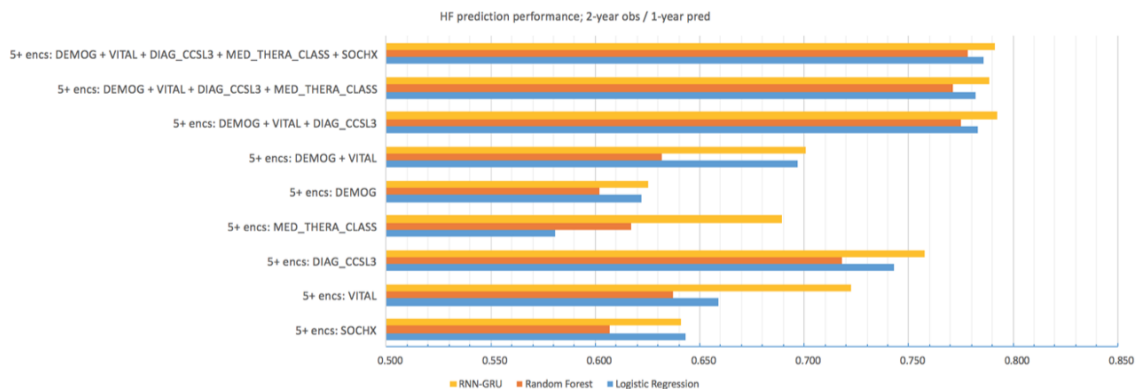


Figure 3.3: Prediction performance as a function of feature set used. Individual data types (demographics, medications, diagnoses, vital signs, social history) as well as combined data types were used.

the AUCs do not improve. With all data domain, the optimal AUC is from RNN-GRU model (AUC= 0.791), which is slightly better than for LR model (AUC=0.786).

Performance as a function of data density

An estimate of the data density in the observation window is computed for each case and control by first accounting for each instance of a data domain along with the associated date, and the number of distinct dates for each patient is computed and is taken as the estimate of density for the patient. The observation window was set at 2 years. Figure 3.3 shows number of patients that had a number of distinct encounters. A total of 4.8% of cases (red line, Figure 3.4) and 3.5% of controls (blue line, Figure 3.4) had less than 5 encounters. Half of the controls have less than 20 encounters and half of the cases have less than 25 encounters.

Figure 3.5 shows the prediction performance as an effect of the data density. To do this, all models (RNN-GRU, RF, LR) were trained on sets of data with different data densities. RNN, RF and LR models are trained on datasets with increasing cut-offs for number of minimal encounters for each patient in increments of 5, from 0 to 70, where all patients are used in the lowest data density category (0 and above). For RNN models, 5 iterations of the model are run on the dataset for each data density. For LR and RF models, 5 iterations

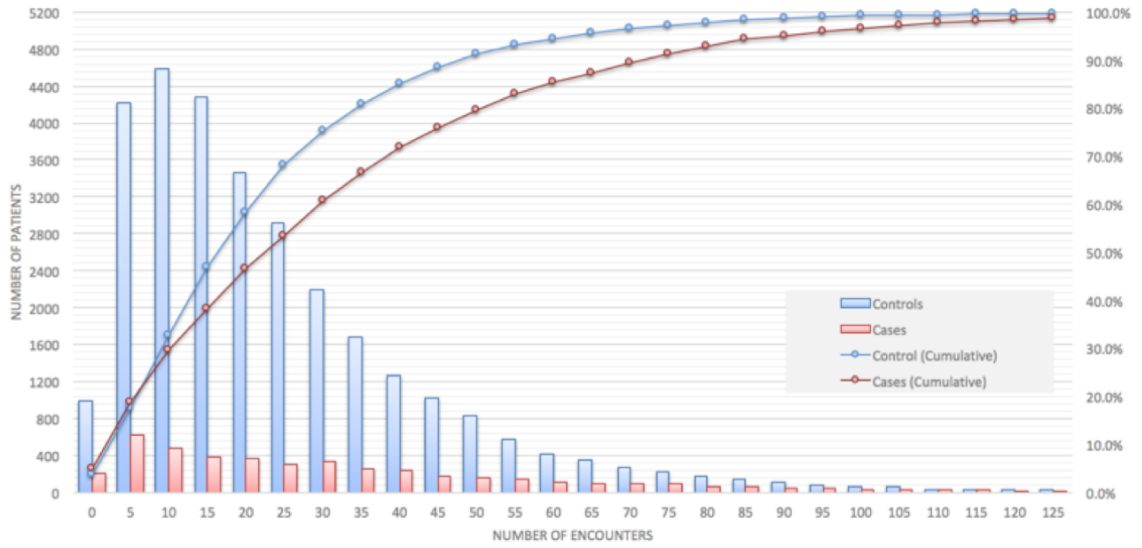


Figure 3.4: Distribution of the number of encounters in the 2-year observation window for cases and controls.

of 10-fold cross validation are run on the dataset for each category of data density and the average AUC is reported. In Figure 7, the bar graphs show that as the density threshold increases there are fewer cases (red bars) and controls (blue bars) selected as training dataset. The blue, green and red curves in Figure 3.5 shows the mean of AUC as a function of the data set density (horizontal axis).

The prediction performance peaks at 5 encounters or more to 10 encounters or more, then quickly declines with increasing minimal number of encounters. For LR and RF models, the optimal AUC is about 0.78, and LR slightly outperforms than RF when minimal number of encounters is less than 10, and RF models performs better than LR after minimal number of encounters reaches 30. The similarly pattern is shown for RNN model, moreover, after minimal number or encounters reaches 40 (AUC for RNN=0.682, and 0.733 for LR), the performance of RNN model declines faster than traditional models.

Performance as a function of data set size

Performance was measured as a function of data set size, ranging from 10% to 100% of

the total dataset (Figure 3.6). All patients (4,370 cases and 30,132 controls) were included in this analysis and all data domains were used. For each data size, the patients are randomly selected from all subjects, For LR and RF, 10-fold cross validation was performed on the entire set of patients. For RNN, the data set is randomly split into 75% training set and 25% testing set.

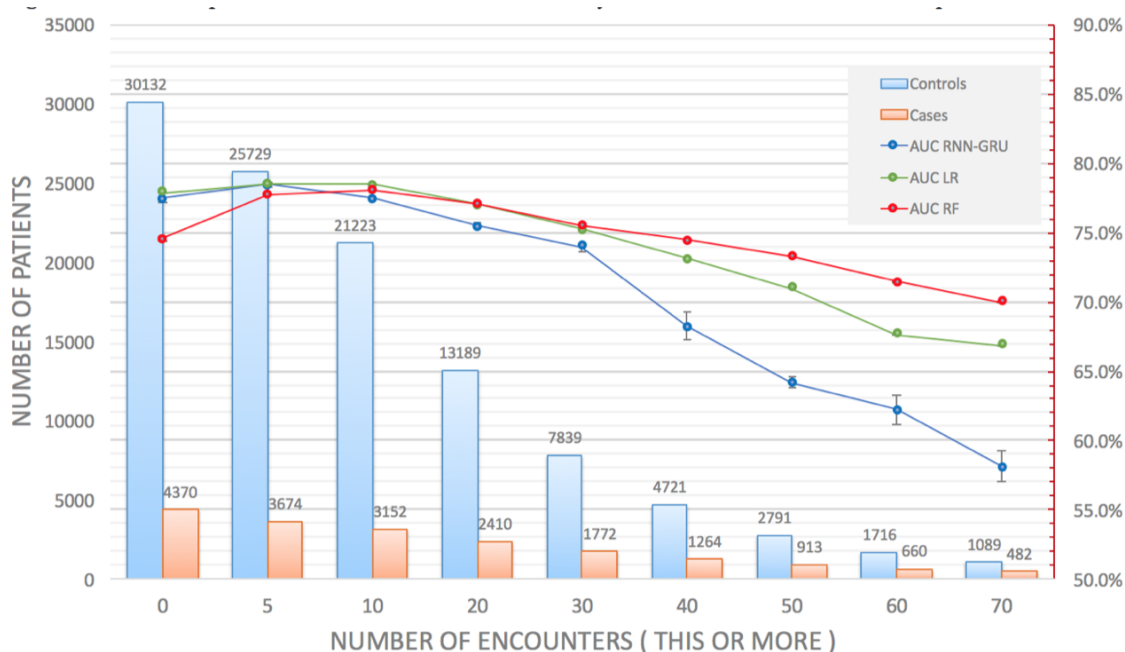


Figure 3.5: Prediction performance as a function of data density. Values are recorded as AUC of prediction.

As the data set size increases, the prediction performance for all models improve substantially, particularly for LR and RNN models. For LR, performance increases rapidly until at least 40% of the dataset (13,800 patients) is used, and the increase reaches plateau at 70% or more of dataset being used. For RF, the performance reaches plateau after 40% of data, and improvement of performance is not as rapid as shown in LR model for 40% or less of dataset is used. For RNN, performance continually increases even until 100% of the dataset is used. It is interesting to note that RNN performance does not surpass performance of LR and RF until 100% of the dataset is used.

Performance as a function of data domain and data set size

To further assess the model performance in relation to number of features in the model and training set size, we select two data domains, one of which contains fewest number of features (vitals, 15 features and one contains the moderate number of features (medications, 93 features), and run models by varying training set size. We compared the model performance to those using all data domains (Figure 3.6).

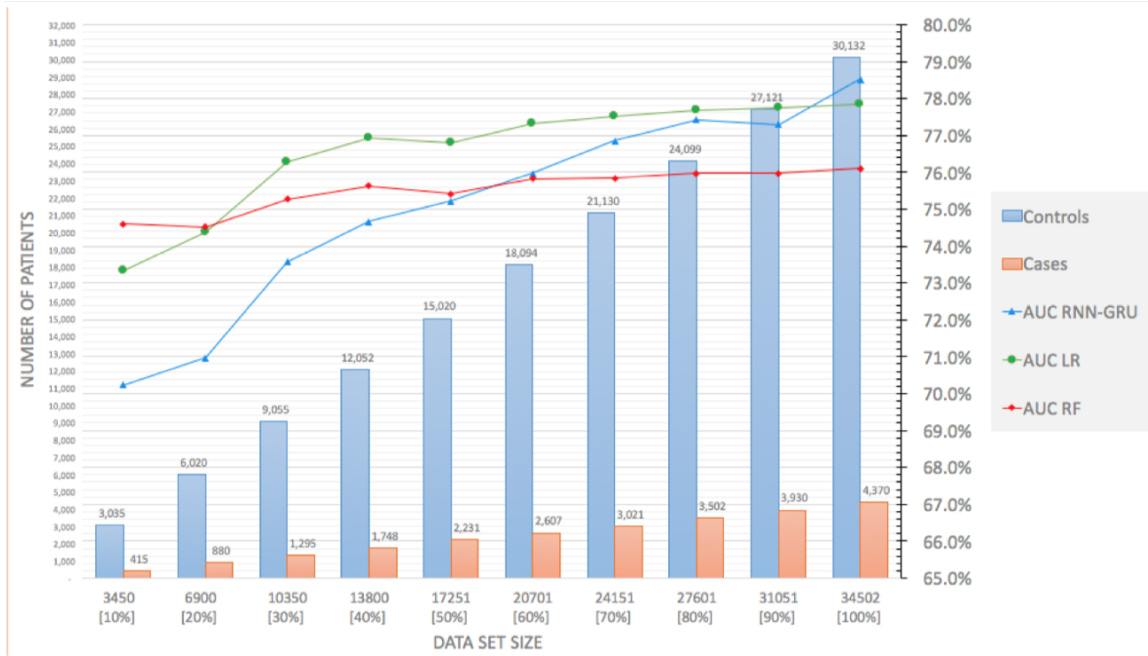


Figure 3.6: Prediction performance as a function of training set size. Values are recorded as AUC of prediction.

Using only vital signs, RNN is superior to LR, and LR is superior to RF model. The performance shows significant increase until 30% of dataset is used for LR and RNN models, and flats out after it, while for RF model, the model performance continues increasing till all data is used (Figure 3.7).

In medication data domain (Figure 3.8), RNN model still performs best, followed by RF model, then LR model. For RNN model, the performance increases rapidly until 30% of data is used, and increases slightly afterward. For RF, the performance only increases slightly with increasing data set size, from 0.62 to 0.63, while for LR, the performance

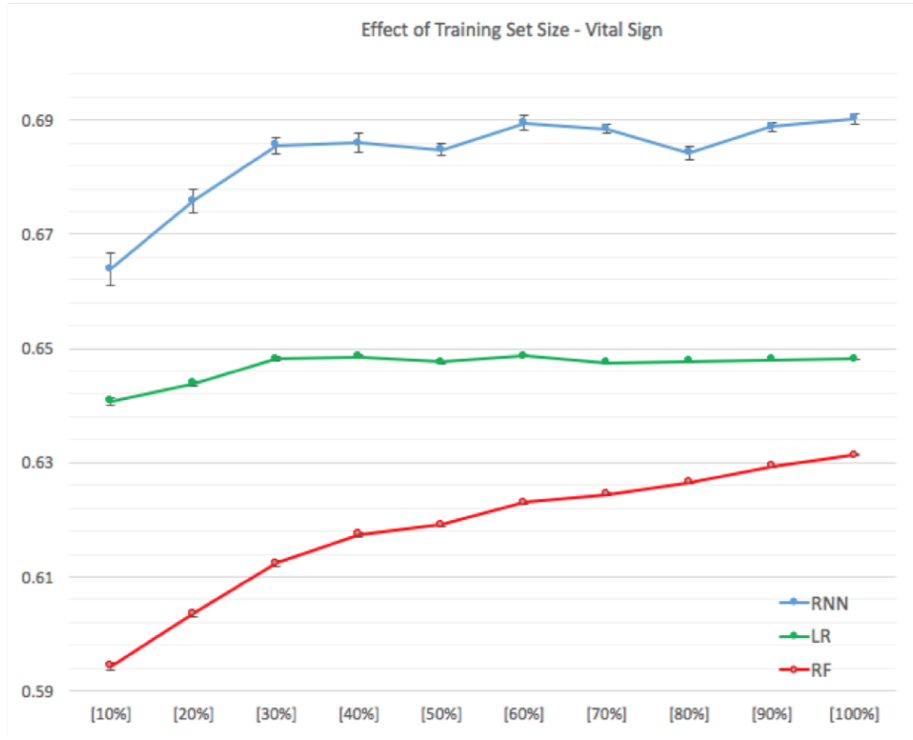


Figure 3.7: Prediction performance as a function of training set size, using only the vital signs as features.

continuously improves with increasing data set size, from 0.55 to 0.588.

Performance as a function of prediction window length

The impact on model performance was evaluated for a window length of 180 days to 2 years (Figure 3.9) with a fixed two year observation window for all data domains. As expected, performance for RNN, LR and RF classifiers declines in relation to increasing prediction window length (i.e., the time before the actual clinical diagnosis). For the RNN model, prediction performance is at or above 0.775 AUC for prediction windows less than or equal to 1 year. Performance declines rapidly for prediction window lengths longer than 1 year, and AUC is 0.753 at 2 year prediction window

Performance as a function of observation window length

The impact on model performance was evaluated for a window length of 90 days to 3 years (Figure 3.10) with a fixed one year prediction window. As expected, performance for RNN, LR and RF classifiers increases in relation to increasing observation window length.

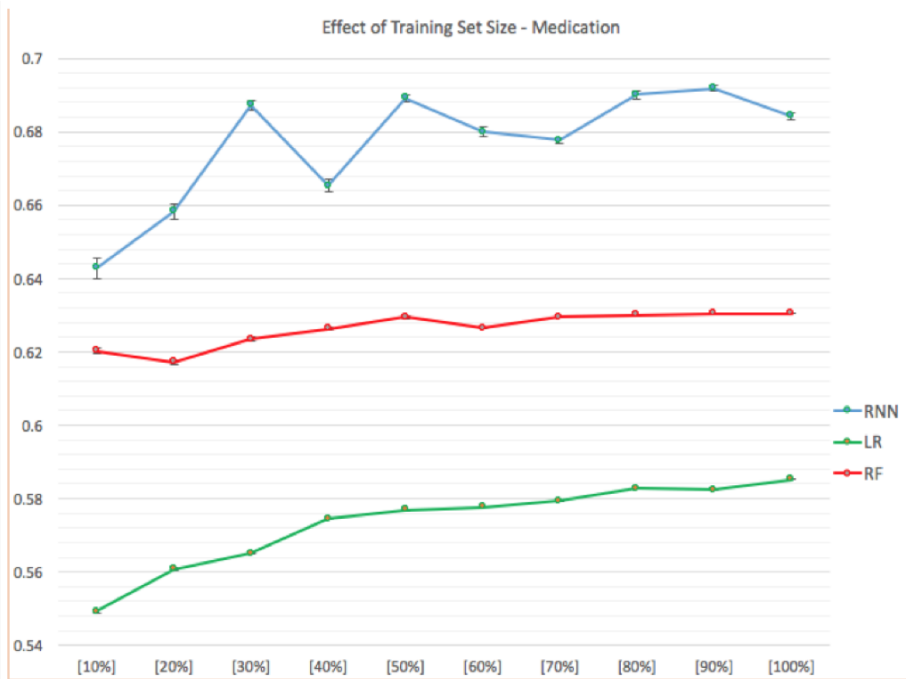


Figure 3.8: Prediction performance as a function of training set size, using only the medications as features.

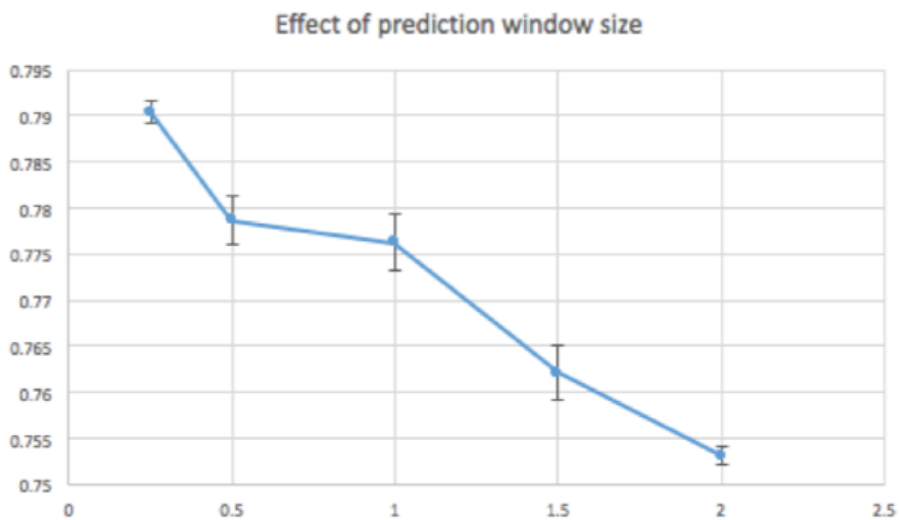


Figure 3.9: Prediction performance for RNN model as a function of prediction window size. Values are recorded as AUC of prediction.

Prediction performance is at or above 0.77 AUC for observation windows greater than or equal to 1 year. Performance does not improve much varying observation window from 1 year to 2 years, but significantly improved for 3 years observation window (AUC=0.795).

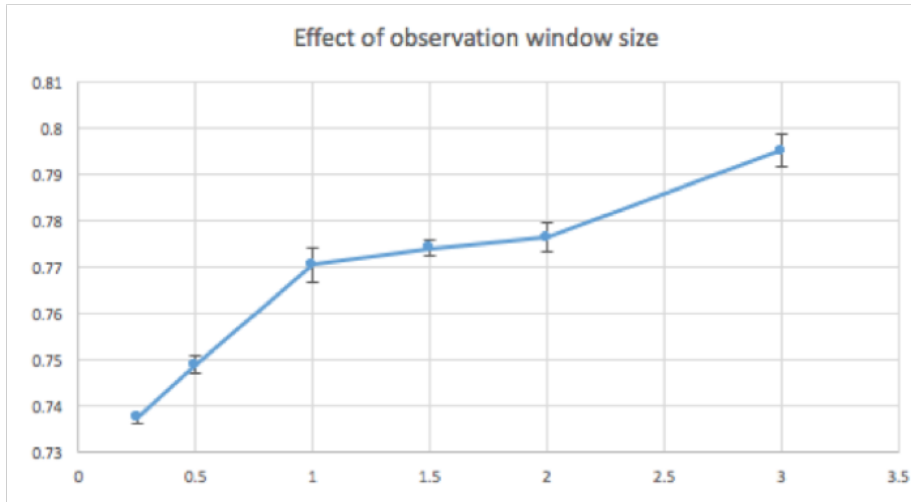


Figure 3.10: Prediction performance for RNN model as a function of observation window size. Values are recorded as AUC of prediction.

Discussion

The adoption of electronic health records in US healthcare is contributing to an explosive growth in diverse longitudinal patient data that will increasingly support population level analyses to support individualized patient care. Predictive analytics represent one important domain that is relevant to early detection of disease, avoidable care (e.g., hospital admission), and clinical decision support, among numerous other applications. The quantitative performance of a predictive model, represented by the AUC, is often a dominant focus of concern, but this represents only one of several factors that will dictate the overall utility of and the likelihood that a model will be adopted for clinical care. Further, while previous attempts to explore how the quantity and diversity of EHR data influences model performance (e.g., with respect to AUC), we sought to understand the implications these sensitivities have with respect to recurrent neural network models. This is highly significant due to the largely increasing interest in the application of deep learning to healthcare.

Our analysis is consistent with previous work [42, 43], in that we show that model performance is strongly influenced by the diversity of data, basic feature construction, and observation and prediction window sizes. We provide additional insight that concerns recurrent neural network models in particular. Namely, we showed that recurrent neural networks provide substantial improvement in performance gain compared to traditional machine learning models (logistic regression and random forest), especially when the data are sparse. Furthermore, from the results it can clearly be seen that RNN significantly outperforms LR and RF in predicting HF early. Thus, it is important to note that temporal sequences of events carry important value in early detection of heart failure. While we have shown that RNN (with GRU architecture) exhibits superior performance to traditional machine learning models such as logistic regression and random forest, it is important to note the potential limitations of our current implementation of the GRU model. In our implementation of the GRU model, patients are modeled as one-hot encoded vectors of all feature types (e.g., vital signs, demographics, diagnoses, medications, labs) concatenated, and the GRU is learned on this concatenated representation of raw features. However, it should be noted that the hidden layer representations can be learned for each feature type independently, in which the learning task is modeled as a series of successive GRU models (e.g., learn h on the demographics features, then feed the resulting $y_{demographics}$ (a set of weights for each patients predicted class) into a GRU taking as features the one-hot encoding for vital signs, then feed the resulting y_{vital} (set of weights for each patients predicted class after learning separate GRU hidden layers on demographics and vital sign features, in 2 successive GRUs) into a GRU taking as features the one-hot encoding for diagnosis features, etc. With respect to the effect of observation and prediction window size on model performance, our results show that RNN models exhibit similar trade-offs in performance with regards to window sizes, as traditional machine learning models (LR and RF). In the RNN model, as with other models, a longer observation window increases the volume of data and also reduces the number of patients to whom the model could be

applied in a given provider group. Furthermore, following the same trend seen in LR and RF, RNN model performance improved substantially with increases in observation window length, up until a period of two years after which plateaus with increases in observation window length. Performance began to plateau for observation windows longer than two years. While it is important to note that many patients do not have substantial data outside of the 3-year window before HF diagnosis, further exploration can help to elucidate potential pathophysiological processes for HF which may play an important role in HF development during the 3-year window preceding HF diagnosis. The most surprising finding in this study is that the RNN models can lead to substantial performance gains over traditional machine learning models (LR and RF) when the data domains used are sparse in nature (i.e., very small number of features). It should be noted that training set size was tested on data domains for vital signs and medications, in order to further understand the effect of training set size limitations of the RNN model. Vital signs and medications data domains are of lower dimensionality than the aggregate set of all data domains, which allows for analysis to understand at which training set size performance begins to plateau. In our analysis of training set size on these sparse data domains (vital signs and medications) we show that there is an optimal training set size where optimal AUC performance can be reached, which is 30% of the entire dataset (i.e., increasing training set size beyond this point does not lead to substantial increases in AUC. However, there is no definitive evidence that there would be a significant increase in performance when increasing the training set size beyond 30%. This result shows that 1) for RNN models, an optimal training set size may be reached, 2) for the specific application of early detection of heart failure, a training set with greater than 34,502 patients may be required in order to reach optimal performance. This is an important finding and plays a role in future consideration of adopting predictive models in clinical settings. Future work involving using RNN models for early detection of diseases should carefully observe the impact of training set size on model performance, and this threshold of training set size, which will likely be different depending on the clinical ap-

plication, should be determined before clinical adoption. There are plenty of directions for future improvements to this work. The future improvements can be structured into one of two categories: 1. Improvements to the medical experimental design, 2. Improvements to the machine learning methodology. In terms of improvements to the medical experimental design, it is important to note that we only used outpatient data for the analysis. This specific clinic from which the data were sourced did not have inpatient encounters. However, previous work [42, 43, 44, 50], previous work suggests that grouping medical concepts in terms of ontologies can lead to substantial performance gains[44, 43]. Ng et al. observed that construction of features using ontologies for ICD-9 codes and medication classes can lead to improvement in model AUC, and can lead to reduction in the number of features, thus aiding interpretability. It has been suggested that curation of data using ontologies such as CCS codes can substantially reduce data diversity without loss of information, and in some instances result in gain in information. Thus, it is not unreasonable to postulate that leverage existing ontologies can further improve performance for RNN models. Regarding improvements to the machine learning methodology, future work should focus on more sophisticated deep learning models to capture temporal relationships between events. In particular, differences in time between successive events should be modeled adequately – more frequent encounters over a period of time may indicate a more complex or sick patient. It is important to understand the potential for adoption of predictive models for heart failure treatment. In the context of providing routine surveillance in order to identify patients with high risk of developing HF, is of tremendous value to health systems. However, adoption will depend, in part, on the data-specific requirements such as data diversity and ad data density. Furthermore, adoption is more favorable when similar model performance can be achieved with fewer types of data. Our analysis on the impact of training set size on RNN model performance using only vital signs and only medications as data domains, has shown that there is a point, for any data domain used in model training, where there is an optimal training set size that yields an optimal performance (i.e., if training set size is

increased beyond that point, the performance generally plateaus). For example, when training set size is increased beyond 20% of the total data set size in terms of number of patients, the prediction AUC hovers around 0.70. Regarding data requirements that may influence future clinical adoption, in our analysis, RNN model performance was considerably better when using either diagnoses or vital sign data either alone or in conjunction with other features, compared to other data types (i.e., demographics/health behaviors, medications, social history). This trend can be witnessed across all model types. It should be noted that other data sources such as behavior data (e.g., those which could be gathered from wearable devices), imaging data (e.g., echocardiogram), and lab data (e.g., hemoglobin A1c values) may add additional value, although they were not extensively used in this study. Nevertheless, our study shows clearly that recurrent neural networks are superior to existing standard machine learning approaches in early detection of heart failure. Furthermore, our study shows that all data types are informative for early detection, with diagnoses and vital signs being the most important. Our work should motivate future efforts for clinical adoption of predictive models for heart failure.

Conclusions

We illustrate that recurrent neural networks outperform traditional machine learning models in early detection of heart failure. Further, we show that there are additional benefits of recurrent neural network models for situations where the data domains used are sparse in nature. Our work should motivate future efforts to understand important advantages and limitations of recurrent neural network models in the broader effort of pushing for clinical adoption.

CHAPTER 4

MODEL DEPLOYMENT

4.1 Tools for Model Deployment

We describe two novel toolkits for deployment of predictive models. The first is a FHIR-based web service for conducting computational phenotyping in large patient cohorts. The second is a cloud-based predictive modeling system leverage Amazon Web Services Elastic MapReduce clusters for efficient model building and results analysis.

4.1.1 FHIR-Based Web Services for Computational Phenotyping from Electronic Health Records

Summary

Phenotyping from electronic health record (EHR) data can play an important role in informing clinical decision making, but current phenotyping algorithms require human supervision and are not interoperable between different data formats. We implemented a web service that computes phenotypes automatically after a user uploads EHR data following the standardized FHIR data model. Our proof of concept should motivate further efforts to develop interoperable platforms for phenotyping.

Background

Adoption of electronic health records (EHRs) by hospitals and physicians has increased dramatically since 2008 as a result of the federal governments HITECH program.[51] A major goal of Stage 3 of this program is improved clinical decision making. This is hindered partially by the difficulty of extracting meaningful concepts, or phenotypes, from EHR data. [52] Current approaches to phenotyping often involve expert specification, and

can be labor intensive. Despite much research [53], a standardized platform for phenotyping using large databases of patients is still lacking. [54, 55] One major obstacle is the lack of a practical and easily implementable standard clinical data model. The Fast Healthcare Interoperability Resources (FHIR) Specification [56, 57, 58] is gaining rapid acceptance as such a standard. [59] To facilitate interoperability, scalability and ready access to our phenotyping algorithm, we implement it as a web service that takes as input patient data represented as FHIR resources, calculates phenotypes in the back end, and displays results via an API and web application.

Methods

EHR data is first converted to the standardized FHIR data model which is specified as a set of resources. Three FHIR resources are used for the phenotyping algorithms: Patient, Admission, and Medication. The FHIR files are stored in a JSON format and can be constructed from EHR data such as hospital admissions, diagnoses, medication orders and general patient information. The web application allows users to upload patient data mapped to FHIR resources and to specify which phenotyping algorithms to run on the uploaded data. A phenotyping algorithm running in the back end is then applied to the uploaded patient data. Currently, we implement a set of algorithms from PheKB [8] which automatically computes phenotypes using a rule-based algorithm based upon diagnostic and medication events. Additional algorithms may be implemented as plug-ins to the back end, allowing for flexible usage. The API returns phenotyping results as another JSON file, which is displayed to the user via a web application (see Figure 4.1). As a proof of concept for the web service, we built a web application hosted on Amazon Web Services, a cloud computing service, which allows users to upload patient data in the form of FHIR files and to run a phenotyping algorithm based upon user-specified parameters. The web application reports to the user a list of phenotypes computed from the uploaded dataset. To test the system, we first converted data for a subset of 1,000 patients from the Multiparameter Intelligent

Monitoring in Intensive Care II (MIMIC-II) database [9] into standardized FHIR files. We uploaded the files onto the web service, choosing to run a phenotyping algorithm which we implemented. After this step, back end algorithms are triggered by the web interface to compute phenotypes. The current implementation of the algorithms require diagnoses to be coded using the ICD-9 scheme. Further, medications are used in the uploaded format without further processing. The results of the algorithm were displayed to the user after the algorithm execution finished. Other than the initial data upload and algorithm choice step, all functions of the software are fully automated and require no human supervision.

Results and Discussion

This web application provides a convenient way for physicians to upload patient data and quickly discover patient phenotypes in the dataset. The process involves minimal human supervision apart from the data uploading step. This web application has practical significance, as it takes input data following a standardized format, addressing pressing concerns with interoperability between EHRs. In a future version we plan to improve the analytics back end and the web interface by allowing users to specify phenotyping input parameters and view phenotypes for specific patient subgroups. Furthermore, while we address the issue of structural heterogeneity by using the standardized FHIR data model, we plan to address terminological heterogeneity by aggregating data for similar events into more interpretable classes (e.g., combining similar diseases or medications into classes before calculating phenotypes). To facilitate the end goal of improved clinical decision-making, we plan to incorporate machine learning algorithms into the back end for use cases such as risk prediction for life-threatening conditions (i.e., sepsis or myocardial infarction) based upon phenotypes. While we currently focus on the usage of phenotyping in the ICU setting, the web service is scalable to large datasets and has the capacity to incorporate a wide variety of phenotyping algorithms and use cases.

Conclusion

We have built a working prototype of a computational phenotyping web service. We successfully converted an EHR dataset into standardized FHIR files, uploaded the data via an API, and ran a phenotyping algorithm on the uploaded data. Our API provides the results of the phenotyping algorithm to a web application for viewing by the end user.

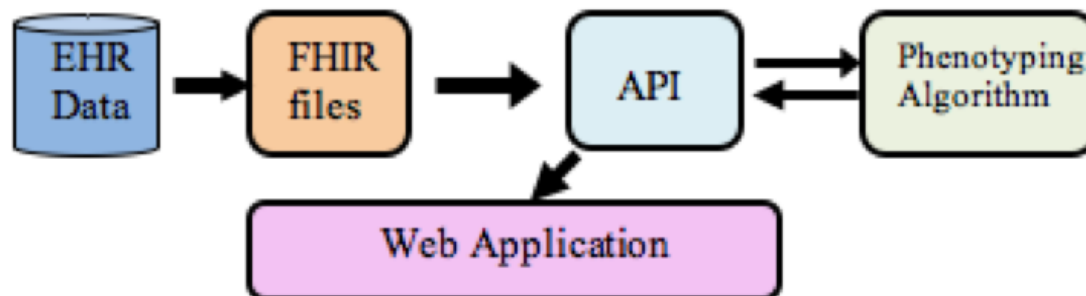


Figure 4.1: An illustration of the general framework for the phenotyping web service.

4.1.2 Cloud-based Predictive Modeling System and its Application to Asthma Readmission

Prediction

Abstract

The predictive modeling process is time consuming and requires clinical researchers to handle complex electronic health record (EHR) data in restricted computational environments. To address this problem, we implemented a cloud-based predictive modeling system via a hybrid setup combining a secure private server with the Amazon Web Services (AWS) Elastic MapReduce platform. EHR data is preprocessed on a private server and the resulting de-identified event sequences are hosted on AWS. Based on user-specified modeling configurations, an on-demand web service launches a cluster of Elastic Compute 2 (EC2) instances on AWS to perform feature selection and classification algorithms in a distributed fashion. Afterwards, the secure private server aggregates results and displays them via interactive visualization. We tested the system on a pediatric asthma readmission task on a

de-identified EHR dataset of 2,967 patients. We conduct a larger scale experiment on the CMS Linkable 2008-2010 Medicare Data Entrepreneurs Synthetic Public Use File dataset of 2 million patients, which achieves over 25-fold speedup compared to sequential execution.

Introduction

High healthcare costs have placed a burden on the federal budget in the United States. Over 90% of Medicare expenditure can be attributed to the management of chronic diseases[60]. However, the quality of care is still far from optimal for many patients, especially those with chronic conditions such as asthma[61]. While there has long been an interest in lowering asthma readmission rates, most predictive modeling studies for asthma have applied a small number of models and may be limited by small datasets. Fortunately, the rapid adoption of electronic health records (EHRs) in healthcare systems provides an exciting opportunity for researchers to leverage this data for secondary uses such as predictive modeling.

While predictive modeling approaches can aid in the detection of readmissions, the predictive modeling process is tedious and time consuming. Researchers often evaluate many models and compare performance metrics between them. Each model may involve different cohort selection criteria, or different features used in predictive modeling tasks. Furthermore, researchers may elect to evaluate several different algorithms in order to choose the best method for predicting a particular target outcome. These iterative predictive modeling efforts will accumulate and lead to large differences in performance metrics attained when comparing the outcomes of different models. Furthermore, with the tsunami of EHR data we need a more scalable computing infrastructure. Taking the aforementioned drawbacks together, we argue that the traditional predictive modeling pipeline is in need of a major overhaul.

With the rapid adoption of EHR systems in hospitals, predictive modeling will be of major interest in the clinical setting. A number of studies have performed predictive mod-

eling for applications such as asthma readmission prediction in hospitals[62, 63, 64, 65]. However, most of these studies were done using either standalone software products for statistical analysis, or computer code written independently by researchers. Such approaches are often conducted entirely on the researchers local computers, and are not scalable with large datasets that are made available as EHR adoption grows rapidly.

Meanwhile, there is evidence that cloud computing can be leveraged in order to support big data analytics on large datasets over a large number of machines in a distributed setting[66, 67]. To date, there does not exist a cloud based web service that supports predictive modeling on large healthcare datasets using distributed computing. There have been some implementations of predictive modeling software. For example, McAulley et al. built a standalone application for clinical data exploration and machine learning[68]. However, the tool was run on local machines and was not deployed on the cloud for easy use by others. The lack of development of health analytics systems on the cloud may also partially be due to the concern of privacy and security of patient data on the cloud.

In addition to the problem with large datasets, researchers often run many iterations of predictive modeling studies before arriving at a desired result. Each iteration may involve changes in the study cohort, features used, and specific machine learning algorithms run. Constantly toggling these parts of the process is tedious and may result in errors. Ng et al. developed the PARAMO system, a predictive modeling platform which constructs a large number of pipelines in parallel with MapReduce/Hadoop[69]. However, PARAMO is built on the users own cluster, which is not always available in every clinical institution, and also lacks scalability when faced with large datasets beyond the capacity of their existing cluster. In addition, most pipelines such as PARAMO are difficult to deploy in a clinical setting due to the large expenses required to maintain servers. Therefore, these systems make little to no impact on clinical decision-making.

To help address the limitations of current predictive modeling pipelines, we developed and deployed a hybrid system that combines a secure private server with the cloud-based

Amazon Web Services (AWS) Elastic MapReduce platform. The system consists of a web service that runs on a private server in a secure environment for preprocess patient data into feature matrices, and an on-demand AWS web service to perform predictive modeling computations. Note that such a hybrid setup enables security of the patient data and at the same time leverages the scalable computing infrastructure on the cloud. Our system is highly customizable to support various predictive model configurations. Furthermore, the system is highly scalable, as the number of cloud-based machines launched can increase with the size of input data. Finally, the system is cost effective because the AWS Elastic MapReduce cluster is only launched when predictive modeling jobs need to be run.

We applied our platform to a predictive modeling task of identifying patients at risk for asthma readmission using a cohort of patients from the Childrens Healthcare of Atlanta EHR system. As one of the most common chronic illnesses in children, Asthma costs over \$56 billion each year, placing a financial burden on the healthcare system[70]. Asthma affects 7.5 million children in the United States annually, and leads to a total of 10.5 million missed school days each year[71]. However, a child whose asthma is properly controlled with education, medication and lifestyle has a better chance of avoiding emergency department (ED) visits as an inpatient. In addressing these issues, care managers want to understand the trends and patterns in the entire patient population. Currently, that is done by grouping patients using diagnosis categories such as Clinical Risk Groups (CRGs) [72] or by risk stratification using risk scores such as clinical risk scores (CRS) for asthma. However, neither approach provides homogeneous patient clusters for purposes of determining targeted treatment protocols. As an alternative, we propose to use our predictive modeling system based on a machine learning strategy to identify patients at high risk for readmission from context-specific information.

In addition to the asthma readmission prediction task, we showed that our system is scalable to large datasets by successfully running a prediction task on the publicly available CMS Linkable 2008-2010 Medicare Data Entrepreneurs Synthetic Public Use File (DE-

SynPUF) dataset, which includes over 2 million patients.

Methods

In this section we describe the design and implementation of our predictive modeling system.

General Overview

The system is composed of two main elements, a persistent web service in a private environment such as a hospital's internal server and an on-demand web service on a cloud-computing environment such as AWS. The persistent web service constantly runs on a dedicated server, houses raw EHR data, and performs preprocessing of data. Processed data in the form of event sequence files are sent to the predictive modeling module, which is powered by an on-demand web service such as an AWS Elastic MapReduce cluster. Once the predictive modeling module is finished, the results are aggregated and displayed to the user in the performance analysis module running on the persistent web service. Figure 4.2 illustrates the key components of our system.

Preprocessing

Data from the EHR are first converted into event sequence files to be used as input to the predictive modeling module. The event sequence files are in the form of text files where each line represents one distinct event from the database. Each record is represented by a tuple in the format (patient, event, timestamp, value), where patient, event, timestamp and value represent the patient ID number, event name, date and time of the record, and a value for the event, respectively. In the case of binary events, such as medication and diagnostic events, the value for a tuple is set equal to 1. In the case of events that are normally associated with numerical values, such as lab values, the value for the tuple is set to the numerical value that is present in the record. In the case of categorical events, such as

gender recorded in an admission, the value for the tuple is set to the alphabetical value (i.e. F and M for gender) from the record. The final event sequence files are used as inputs to the predictive modeling module. Instead of using the raw information such as actual patient IDs and ICD-9 codes, the data in event sequence files are further transformed into internal coded values on the persistent web service before being uploaded to the on-demand AWS web service. In particular, patient IDs and event names can be hashed into internal IDs. Thus, the raw patient information would not be used in the predictive modeling processes running on the cloud for security considerations. After the predictive modeling module is finished, specific patient ID numbers and event names may be decoded on the persistent web server before running the performance analysis module on the dedicated server.

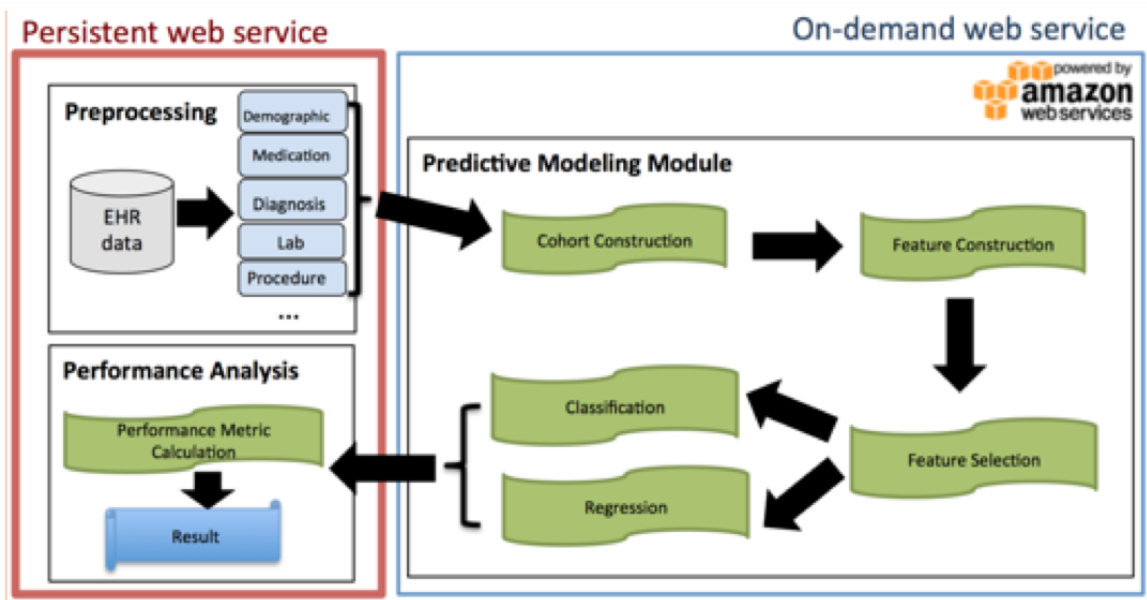


Figure 4.2: An overview of our predictive modeling system. We extract EHR information from the database from the hospital and store the information in these event sequence files through persistent web services running on a private dedicated server. These event sequence files are uploaded to the web service on the cloud.

Predictive Modeling Module

The predictive modeling module consists of several stages. First the cohort construction and feature construction stages are conducted. Next, cross validation stages comprised of feature selection followed by either classification or regression are run. We call a concrete step in one predictive modeling process a task. Examples of tasks include constructing diagnoses features, or building a logistic regression classifier on a specific training set. We organize all those tasks into different computation stages: cohort construction, feature construction, cross validation data splitting, feature selection, classifier training and classifier testing. Note that each stage corresponds to one or many computational tasks. All of these tasks occur on the on-demand web service, which is implemented with AWS Elastic MapReduce. The predictive modeling module launches a new AWS Elastic MapReduce cluster consisting of multiple EC2 instances for each analysis workload. The predictive modeling module aggregates all computational tasks from all stages of predictive modeling process and schedules them to run in parallel on the provisioned AWS cluster. The user is allowed to choose the number EC2 virtual machine instances and the types[i] of machines to use in the MapReduce cluster. Next we introduce those computation stages in more details. Cohort construction: Once event sequence files are uploaded to the on-demand web service, the user can specify a set of filtering criteria that can be used for constructing cohorts (Figure 4.3 A). A user can define the patient selection criteria for cases and controls. For example, for an asthma readmission prediction study, the user may define the case patients by inputting readmission as the target event to predict. Patients who have the target event will be regarded as cases and others as controls.

After user defines the target event to predict, the user can further narrow down the cohort to study by specifying conditions. For example, user may require all patients in cohort should have at least 3 inpatient events. After the user selects case cohort inclusion conditions, the user may want to balance the number of cases and

controls via matching (Figure 4.3 A). In this situation, the user may elect to select a

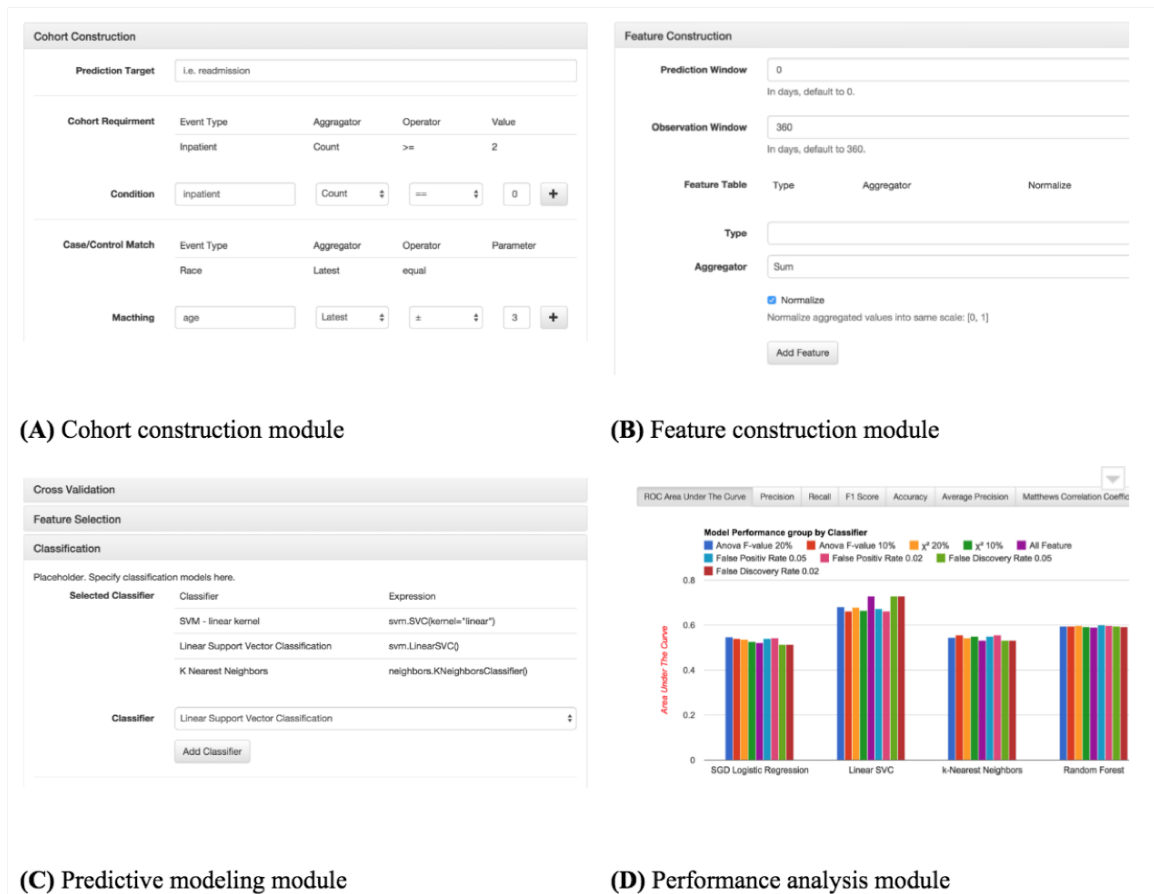


Figure 4.3: Screenshots of the cohort construction module (A), feature construction module (B), predictive modeling module (C) and the performance analysis module (D). The cohort construction module allows users to specify criteria for selection of cases and for identifying matching controls. In the predictive modeling module, the user may specify parameters for particular feature selection and classification algorithms. The performance analysis module allows users to visualize key performance metrics as well as top predictive features selected in feature selection

limited number of matched controls using a matching algorithm based upon patient similarity metrics or propensity scores. Currently, a simple case-control matching algorithm is implemented, which selects control patients that have identical or numerically similar values for the matching criteria as the case patients.

Feature construction: Features from the event sequence input data may be aggregated depending on user preferences and study design. The feature construction module allows the user to specify a method of aggregation for each event type with respect to its values for each patient. The aggregation methods include mean, sum, count and latest. Mean and sum are the mean and sum of the values across all certain events for a given patient. Count is the number of times the feature occurs as an event. Latest is the value for the feature at the most recent occurrence. Figure 4.3 B shows a screenshot of the feature construction module. During the feature construction phase, besides the feature matrix, other entities will also be computed and stored for later usage. For example, feature value statistics will be used for the performance analysis module on the persistent web service. Feature value statistics we collect include percentage of cases/controls who has certain events and distributions of feature values within cases and controls.

Feature selection: The predictive modeling module runs a combination of feature selection[73] and classification tasks. Since EHR data is often high dimensional in nature, some features may have a large amount of missing or noisy information. Feature selection is used to filter out these features, which should not be considered for predictive modeling[73]. The feature selection algorithms implemented include raw features, the Chi-square feature selection [74], analysis of variance (ANOVA) F-value based feature selection[75], and false discovery rate based feature selection[76]. The raw features method uses all constructed features from the data. The Chi-square method uses a Chi-squared test to evaluate p-values for each feature with respect to the target labels. A smaller p-value indicates a more predictive feature. For example, user may choose to select features with p-values less than 0.05. Likewise, the analysis of variance (ANOVA) F-score method applies an ANOVA test to each feature with respect to the target labels. The false

discovery rate method selects features for being correlated with the target labels using the Benjamini-Hochberg algorithm[76]. Classification and regression: We included all classification algorithms from the Python scikit-learn[48] package, including logistic regression (LR), support vector machine (SVM), k nearest neighbors (KNN), random forest (RF) and beyond. The user is able to specify parameters for each classifier, for example the type of kernel for SVM, the number of trees for RF, the number of neighbors and the distance metric for KNN, and the method of regularization and type of optimization in LR. Similarly, all regression algorithms from the scikit-learn package are also included. Cross validation: In the case of classification or regression problems, cross validation is run for each possible pairing of feature selection method followed by classification or regression methods. The user is able to set the number of folds in cross validation and number of iterations to run cross validation (Figure 4.3B). For each fold of cross validation, feature selection is first run on the training folds of the dataset. The selected features are used in the classification algorithm to be run. The system will figure out the set of tasks to run and their dependencies. For example, on a given fold of cross-validation, classifier C's input depends on the output of feature selector S of the same fold, thus the system runs C after S. For other tasks independent from C or S, the system could schedule them running in parallel with C and S. Parallelization: As described previously, our system runs on an on-demand AWS Elastic MapReduce cluster. The system achieves speed and scalability from two levels of parallelization. The first level is within-task parallelization, which means a predictive modeling task itself is implemented using big data parallel processing techniques. For example, cohort construction and feature construction are implemented using Spark[77]. Thus, different features will be constructed in parallel. Another level is between-task parallelization. All tasks derived from the predictive model configuration form a dependency graph, and the system schedules tasks in topological order. A task will be ready to run whenever its dependencies finish. For example, testing of a classifier could be scheduled to run when the training of the given classifier finishes. All tasks whose dependencies satisfied could

run in parallel. For example, if both classifier A and B depend on feature selector C, A and B can start running simultaneously when C finishes. The degree of parallelization will be determined by the capacity of the cluster, which is usually measured by total number of CPU cores and total memory available. All the scheduling and execution are done using Hadoop and Cascading running on AWS MapReduce clusters. Once all tasks are finished, the system collects and aggregates the results from all clusters distributed file systems. The results are sent to the performance analysis module on the persistent web service. Performance Analysis Module The final module of the system is the performance analysis module on the dedicated server. The results from all classifiers run in the predictive modeling modules MapReduce service are collected, aggregated and stored in a MongoDB database running on the persistent web service. The performance metrics calculated include area under the receiver operating curve (AUC), positive predictive value (PPV), sensitivity, F1 score, accuracy and Matthews correlation coefficient. The web service retrieves the performance metric data from the MongoDB database and displays results to the user in an intuitive interactive interface. Figure 4.3 C shows a screenshot of the webpage displayed to users.

Results

Next we describe the results from the application of our predictive modeling platform to an asthma readmission task.

Asthma Prediction Task Experiment Setup

The study involved a cohort of 2,967 inpatient pediatric asthma patients from the Childrens Hospital of Atlanta (CHOA). There were 1,493 patients who had at least one readmission for asthma treatment, and 1,474 patients who did not have any readmissions. Data for inpatient events representing emergency department initial visits and readmission visits were used. Table 1 showcases general patient characteristics of the study cohort.

To run the preprocessing and performance analysis modules, we hosted the persistent web service on dedicated server running the Ubuntu 14.04.1 LTS operating system, with 24 Intel Xeon 2.6GHz processors (six cores each) and 256 GB of RAM. An on-demand web service was launched in order to run the predictive modeling module. The on-demand web service consisted of an AWS Elastic MapReduce cluster consisting of 1 master m3.medium EC2 instance and 20 slave c3.xlarge EC2 instances. **Feature Construction:** We preprocessed the data set and obtain 5,728 unique patient visits. Each visit has a readmission label showing that whether or not the visit has led to one or more visits within the next 12 months.

We constructed six groups of features: demographic features, diagnosis features, medication features, procedure features, and visit features. The demographic, diagnosis, medication and procedure features are categorical features while the lab and visit features are numeric features. Table 1 shows descriptions and example values for features in each group. We converted the categorical features to 1-over-K binary code representations. For example, the feature race has five distinct categories: White, Black or African American, Asian, American Indian or Alaska Native, and Others. If a patient belongs to the category white, his or her race will be represented by a 5-dimensional feature vector [1,0,0,0,0]. The diagnosis and medication features were binary features, where a value of 1 indicates that an event occurred for that feature and a value of 0 indicates that an event did not occur. Furthermore, we convert the numeric features to z-scores. Taking a 40-month-old patient for example, the z-score of feature age will be -0.91 given that the average and standard deviation of age in our cohort are 100.10 months and 66.13, respectively.

Cohort Construction: We obtained a cohort of 1,493 unique patients with asthma readmission within one year after being discharged and 1,474 unique control patients without readmission matched on age in month and gender[ii]. We use a 1919-dimensional feature vector to represent each patient. We summarize the statistics of demographics of the cohort in Table 2.

Feature selection: We performed feature selection using four separate methods: raw features, ANOVA F-score feature selection, Chi-square feature selection, and false discovery rate (FDR) feature selection.

Classification: We formulated the asthma readmission prediction as a binary classification problem where the two target labels are defined as follows:

We applied four commonly used classifiers: logistic regression (LR),[78] linear support vector machine (linear SVM),[79] K-nearest neighbor (KNN), and random forest (RF)[80]. We used stochastic gradient descent with L2 regularization for the logistic regression, set $K=1$ and use Euclidean distance for KNN, used a linear kernel with $c=1$ for SVM, and used 100 trees for RF. **Performance analysis:** We partitioned the patients into training and testing cohorts in a 3 times 5-fold cross validation process, meaning cross validation was run for 3 iterations. For each fold, we first performed feature selection and then trained the model on the training set (80% of the entire data) using the selected features. Afterwards, we evaluated the model performance on the testing set (20% of the entire data). We used the following evaluation metrics: a) area under the receiver operating characteristic curve (AUC); b) positive predictive value (PPV); c) sensitivity; d) F1 score; e) accuracy. To calculate the final value for each performance metric, we find the mean of the means of each metric across all iterations.

Asthma Experiment Results

Feature selection: Of all of the feature selection methods, the false discovery rate (FDR) method achieved the best overall performance with AUC 0.69, PPV 0.69, sensitivity 0.46, F1 score 0.55, and accuracy 0.77. Table 3 shows the top predictive features selected by the FDR feature selection method in all 10 folds. Six out of the 10 features were verified by pediatric clinicians to be possible indicators for asthma readmission (highlighted in Table 3). Two of the features, the medication fluticasone-salmeterol and the lab total immunoglobulin E (IgE), are known to be strong indicators for asthma readmission. The

fluticasone-salmeterol feature is present in 15% of all cases while present in only 8% of all controls. This result is clinically meaningful because fluticasone-salmeterol is commonly prescribed in more severe asthmatic patients. The total immunoglobulin E (IgE) lab value is 565 IU/mL in cases and 258 IU/mL in controls. This result is clinically meaningful as well, since more severe asthmatic patients tend to have higher values for IgE, a marker indicating sensitivity to allergens[81].

Classification: We performed cross-validation to choose the appropriate number of features that gives the best performance. Cross validation was performed on each possible combination of feature selection algorithm and classification algorithm. For each feature selection method, we collected all features that met the feature selection criteria. These features were used as predictive features in the classification tasks. Table 4 shows the performance of the linear SVM classifier while using different feature selection methods and raw features. The feature selection method with the highest average of all performance metrics was determined to be the one with the best performance.

The results of the four different classifiers using the feature selected by the FDR feature selection method are shown in table 5. There was variability in performance of the classifiers. Linear SVM achieved the highest AUC (0.69). Logistic regression achieved the highest sensitivity (0.99), while random forest achieved the highest PPV (0.89), F1 score (0.74), and accuracy (0.86). It is important to consider these results in the context of the particular application. For the asthma readmission prediction problem, the SVM, logistic regression, or random forest methods may all be considered effective models based upon different use cases. In cases where sensitivity may be important (e.g., detecting high risk patients who may need urgent care), logistic regression may be the best model. In cases where positive predictive value may be important (e.g., when treatment for positively predicted patients is expensive, and financial resource allocation is important), then random forest may be the best model.

System Scalability

To demonstrate the scalability of our system, we ran our system on a much larger dataset, a set of 2.1 million patients from the CMS Linkable 2008-2010 Medicare Data Entrepreneurs Synthetic Public Use File (DE-SynPUF), a publicly available synthetic dataset. It contains approximately 300,000 different kinds of events from the patients. Thus, there were about 2.1 million patients and 300,000 features in the input dataset. The raw event sequence input file size is 6.88GB. We created a predictive modeling workload with more than 3000 tasks. The feature selection and classifier settings are almost identical to those used in the asthma readmission prediction task. The on-demand web service is composed of 10 r3.2xlarge AWS EC2 virtual machines. Figure 4.4 shows the timeline of parts of the task run and the amounts of time spent. the CMS data. The entire running time of the pipeline workflow is about 3 hours. To serve as a baseline, we ran all the tasks sequentially on a single server of the same machine configuration to calculate the total running time of a sequential run. We find that our system achieves a 36-fold speedup over the baseline sequential running time. Note that the feature construction step is only conducted once, while the data splitting, feature selection, model training and model testing steps are done for each iteration of cross validation.

Discussion

We have developed a cloud based system for clinical predictive modeling. Our system is the first of its kind to date, and leverages Amazon Web Services Elastic MapReduce technology to run distributed feature selection and classification jobs in a time efficient manner.

Challenges in Privacy and Security

While the usage of the cloud is relatively new, many users are already using the cloud for hosting personal health information (PHI).[60, 66, 82, 83, 84] In the case that the users

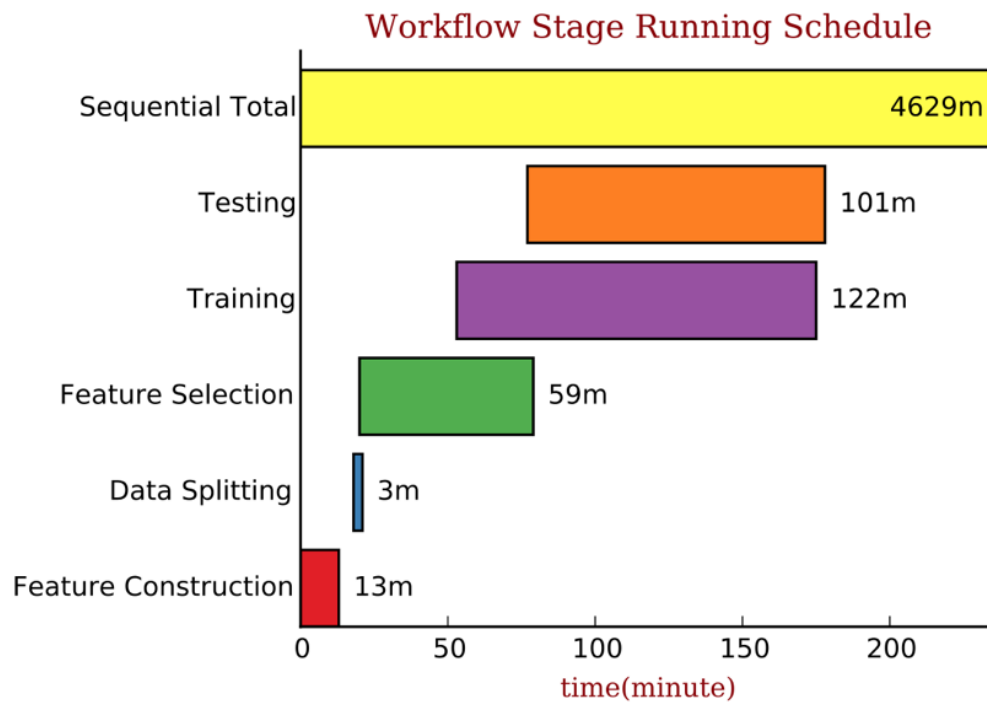


Figure 4.4: Timeline of modules run and elapsed time. The data-splitting, training and testing times refer to the run times for each respective step of cross validation. Times are shown in seconds (s).

are unwilling to store EHR information into the cloud, our system architecture can be used in such a way that preprocessing of data can be performed on the persistent web server, and PHI can be mapped to codes. For example, all information including patient ID numbers and medication, procedure, lab and diagnosis names may be hashed and mapped to different values such that the raw data were not uploaded to the cloud based system. Our system mitigates potential concerns regarding privacy and security of healthcare data. However, we also recognize the heuristic nature of our approach, and in the future we plan to conduct more focused studies on privacy consideration using the cloud in an effort to provide a more theoretical guarantee of privacy.

Conclusion

We have proposed and implemented a hybrid version of predictive modeling system, which combines a private dedicated instance and public cloud computing services. In this system, raw EHR data are converted into standardized features written into event sequence data files via persistent web services on the private server. The de-identified event sequence files are uploaded to an on-demand web service via Amazon Web Services, which subsequently constructs cohorts and features and schedules a series of distributed predictive modeling tasks using big data systems such as Spark and Hadoop. The results of the predictive modeling tasks are collected and displayed to the user in a highly intuitive, interactive user interface on the private server.

We applied our system to a specific task of prediction for pediatric asthma readmission using a cohort of case patients with asthma readmission and matching control patients. The predictive modeling module was successful in the prediction task through a 5-fold cross validation scheme. The system predicted patients at risk for 12-month asthma readmission with an AUC of 0.69.

We plan to improve upon the system by expanding the suite of cohort construction strategies, feature selection algorithms and classification algorithms. Furthermore, we plan

to add functionality for testing multi-class classification tasks (e.g., to be used for detecting multiple types of readmissions).

4.1.3 Executable Data Science Notebook for Phenotype Discovery Pipeline using Observational

Data

Abstract

As phenotype discovery or unsupervised phenotyping becomes a critical component of observational research, the need for transparent and portable phenotyping pipelines has become more pronounced. While standardized representation has begun to emerge for rule-based phenotypes or phenotyping via supervised learning, phenotype discovery remains ad hoc in its methods and representation. Currently, it is extremely difficult to reproduce the phenotype discovered in a scientific papers due to the lack of transparent and easy-to-use analytic pipelines used by the paper's authors. To address this shortcoming, we designed and implemented an open source phenotype discovery pipeline that is represented in a data science notebook environment. The pipeline works with databases in the Observational Medical Outcomes Partnership (OMOP) common data model (CDM) standard. The pipeline can be adapted to any phenotyping algorithm, saves and exports phenotype definitions in a JSON object, and exports a copy of the notebook environment that is easily shared with other users who may wish to either assign patients from their data into the existing phenotypes, or adapt the pipeline to a new dataset with different algorithms or disease conditions. We demonstrated our pipeline in 3 case studies: 1. We perform the heart failure subtype discovery on a heart failure dataset from Sutter Health, 2. Using heart failure subtype definitions learned from Sutter Health, we perform phenotype assignment on another dataset the CMS 2008-2010 Entrepreneurs Synthetic Public Use File (CMS DE-SynPUF), 3. We adapt the existing phenotype discovery pipeline to aid in new phenotype development with different algorithms.

Introduction

Over the past decade, a number of clinical data research networks (CDRNs) have emerged to conduct large-scale research using observational data in a federated manner. One such example is the Observational Health Data Sciences and Informatics (OHDSI) consortium [85, 86], which was formed with the goal of creating a large community for data scientists in the healthcare space to share analysis code and conduct collaborative projects. Alongside the rise of communities such as OHDSI, there has been a concomitant rise in interest in computational phenotyping, which leverages machine learning methods to subtype and characterize patient conditions based on electronic health records (EHR) data. Computational phenotyping has potential for use cases in genomics and precision medicine [87, 88, 89], population health management, and in the understanding of complex diseases such as rheumatoid arthritis [90] and multiple sclerosis [91]. Common practices of phenotyping have included supervised phenotyping algorithms (including both rule-based algorithms as well as classification methods) and unsupervised phenotyping for defining and characterizing phenotypes [92]. While supervised phenotyping is common, as we enter the world of precision medicine, it will be increasingly more important to leverage unsupervised learning algorithms for the purpose of discovering phenotypic subtypes that represent important segments of the patient population. Examples of such unsupervised algorithms include non-negative tensor factorization based strategies [92, 4, 12, 37, 36], non-negative matrix factorization, Gaussian mixture models [39] and topological data analysis [35]. While medical researchers may use the same phenotyping algorithms, their workflows may differ slightly, leading to a difficulty in reproducing each others work. For example, some researchers may extract data directly from a database while others may perform database dumps and load the data in CSV files. Often times, researchers save configuration criteria for their algorithms in scripts, and have multiple scripts that must be run in succession. This poses a difficulty when one researcher desires to share his or her own workflow with another researcher who may not find the workflow intuitive to follow. Finally, researchers

may use different vocabularies for representing phenotypes. For example, medications may be encoded by Generic Product Identifier (GPI) codes in one dataset while they are encoded by RxNorm codes in another dataset. Despite the differences in workflows, it is of utmost importance that researchers collaborate, especially given the highly complex nature of some conditions or patient segments that are poorly understood with respect to patient subtypes. One example is heart failure, which is a highly complex due to patients exhibiting multiple comorbidities [93, 45]. Due to the severity of the disease itself, heart failure leads to high healthcare costs and mortality (i.e., 50% within 5 years of diagnosis) [45, 94]. To address the aforementioned problems and facilitate common practices in phenotyping, we propose to create the shared digital objects for phenotyping, which leverage common data models as input and provide a common computational pipeline to standardize the analytic workflow. To this end, we designed and implemented, a modular and executable phenotyping pipeline in the form of an easily portable data science notebook. The notebook environment has recently gained popularity as a useful tool for efficiently exploring data and performing discovery tasks among data scientists [95, 96, 97], and has been implemented in the form of Jupyter or Zeppelin notebooks, which can be shared online. Our particular notebook is modular and subsections of the pipeline can be swapped out or modified according to the users specific goals. Furthermore, our proposed notebook is integrated with the OMOP data model, and also provides standardized phenotype definitions as output. The phenotype definition is portable in a JSON object, which can be used for assignment of patients to those phenotypes, or for ease of sharing via web services. We present 3 different case studies of phenotyping. The first use case is the implementation of phenotype subtypes of heart failure on a cohort of heart failure patients in the Sutter Palo Alto Medical Foundation clinic, and the subsequent usage of the pipeline with the same configurations for discovery of heart failure subtypes on a cohort of patients from a different dataset - the CMS Linkable 2008-2010 Medicare Data Entrepreneurs Synthetic Public Use File (CMS DE-SynPUF). The second use case is the phenotype assignment to the CMS DE-SynPUF

cohort using the heart failure phenotype definitions obtained from the Sutter PAMF cohort. The third use case is the customization of the notebook with the k-means algorithm for clustering. In all 3 case studies, the phenotype definitions are saved in JSON format and the pipeline and associated system configurations are stored together in a shared directory.

Methods

In this section we describe the implementation details of our phenotyping pipeline.

General Design and Implementation

Figure 4.5 details the general design of our analytic framework. An electronic health record database that follows the Observational Medical Outcomes Partnership common data model (OMOP CDM) is read into the phenotyping pipeline via a database connector. The phenotyping pipeline is implemented in a lab notebook and executes a series of tasks that are required for extraction of phenotypes from the source data: data transformation, phenotype extraction, follow-up analysis, displaying results in-line, and exporting final phenotype definitions and cohorts as well as the discovery and assignment pipeline. The source code for the pipeline and associated instructions for use are hosted on a BitBucket repository at

[https : //bitbucket.org/realsunlab/phenotyping_lab_notebook](https://bitbucket.org/realsunlab/phenotyping_lab_notebook).

Phenotyping pipeline details

The phenotyping pipeline is an interactive coding environment that allows the user to view and execute code to perform all relevant tasks required for extracting phenotypes from source data. The pipeline can be implemented in any lab notebook environment, such as a Jupyter Notebook, or Zeppelin Notebook. In this section we describe all the major steps of the pipeline (Figure 4.6, Figure 4.7, and Figure 4.8).

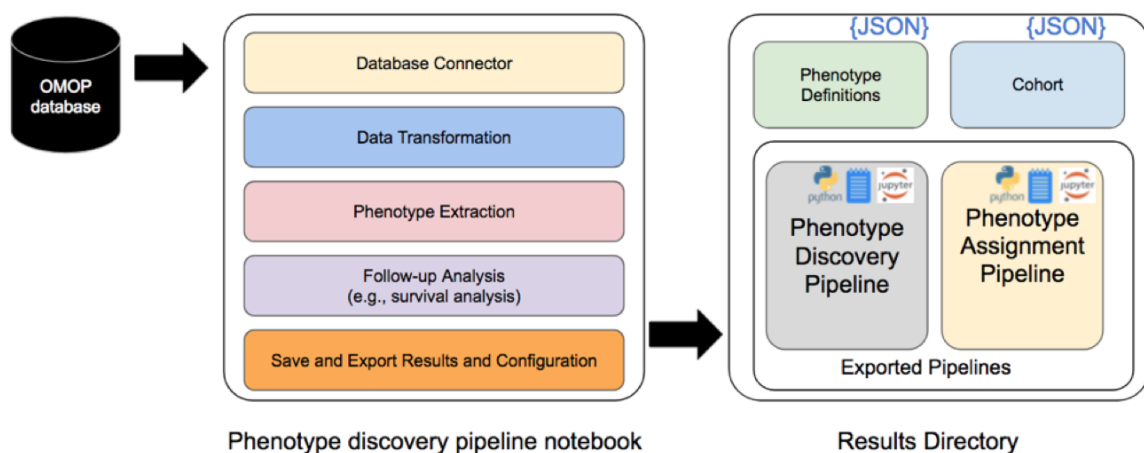


Figure 4.5: Overview of the workflow and phenotyping pipeline components

A

```

connect to database (OMOP)

first we need to connect to a database, and specify the data wanted for phenotyping

• replace database name and login info with your database info
• specify which concepts you want to include in the phenotyping task, these can include for example, medications and conditions
• database should follow the OMOP schema

In [281]: import sqlite3

           # connecting to the database
           conn = sqlite3.connect('/data/db_gutter_omop.db')
           c = conn.cursor()

specify OMOP concepts used for phenotyping

below, please specify the clinical data tables used in the phenotyping task.

• code this as a list of strings, each of which correspond to a table
• for this example we will use the following tables:
  • DRUG_EXPOSURE
  • CONDITION_OCCURRENCE

In [283]: l_clinical_data_tables = ['CONDITION_OCCURRENCE', 'DRUG_EXPOSURE']
           l_clinical_data_date_columns = ['condition_start_date', 'drug_exposure_start_date']
           l_clinical_data_concept_id_columns = ['condition_source_concept_id', 'drug_concept_id']
           l_clinical_data_feature_labels = ['CONDITION', 'MEDICATION'] ## your label for each of the fea

```

Figure 4.6: Screenshots from the phenotyping pipeline notebook. Shown here are specifications for a connection from the lab notebook to a database that follows the OMOP Common Data Model.

B

Phenotyping: display phenotypes

Display phenotypes extracted. Full details of phenotypes are provided in the results folder.

Note:

- If you desire to load the phenotype definitions later (e.g., to project patients from a new dataset onto the definitions), then you will have to load the JSON object
- If you are using a different phenotyping algorithm, you will have to replace this code with something appropriate for handling your algorithm's phenotyping output

```
print 'DRUG_EXPOSURE', '-----'
l_this_mode_descriptions = []
features_this_pheno_idx = np.nonzero(conditions_by_pheno[:, pheno_number_stored_in_datafile])
features_this_pheno_idx = np.nonzero(medications_by_pheno[:, pheno_number_stored_in_datafile])
for item_idx in features_this_pheno_idx: # loop thru the features in the mode, and if they l
    item_name = d_idx_med[item_idx]
    print '\t', item_name
    d_this_item_description = {'description': item_name, 'weight': conditions_by_pheno[item_idx, pheno_number_stored_in_datafile]}
    l_this_mode_descriptions.append(d_this_item_description)
d_this_phenotype_output_JSON['features']['DRUG_EXPOSURE'] = l_this_mode_descriptions

l_phenotype_output_JSON.append(d_this_phenotype_output_JSON)
# re-shuffle nparr to match original ordering in df_X_by_year
l_pts_phenotyped = [d_idx_pt[x] for x in range(num_pts)]
```

```
Phenotype 1 =====
percentage of patients: 10.1 % (n= 390 )
CONDITION OCCURRENCE -----
Atrial fibrillation (NCC)
Congestive heart failure, unspecified (NCC)
Coronary atherosclerosis of unspecified type of vessel, native or graft
Encounter for long-term (current) use of anticoagulants
Other and unspecified hyperlipidemia
Type II or unspecified type diabetes mellitus without mention of complication, not stated as uncontrolled (NCC)
Unspecified essential hypertension
DRUG_EXPOSURE -----
FUROSEMIDE 20 MG PO TABS
FUROSEMIDE 40 MG PO TABS
JANTOVEN 5 MG PO TABS
```

display breakdown of patients

the following code block calculates the number of patients that were assigned to each phenotype and generates a bar chart

```
In [477]: ## show breakdown of patients in phenotypes
l_phenotype_pct_pts = [x['num_pts'] for x in l_phenotype_output_JSON]
l_labels = [x['phenotype_name'] for x in l_phenotype_output_JSON]
plt.bar(range(R),
        l_phenotype_pct_pts)
plt.xlabel('Phenotype #')
plt.ylabel('Number of patients')
```

Out[477]: <matplotlib.text.Text at 0x115cebe10>

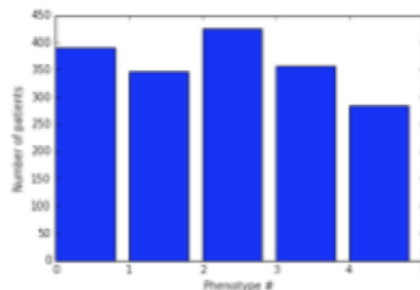


Figure 4.7: Screenshots from the phenotyping pipeline notebook. Shown here are excerpts from the code for running the non-negative tensor factorization based phenotyping algorithm. Descriptions of the phenotypes are displayed in-line. Display of general statistics of the phenotype results are displayed in bar-chart format inline (with Matplotlib).

Follow-up Analysis: Survival

This next section performs a survival analysis on the phenotypes obtained.

What this does:

1. load the phenotypes (saved in a JSON object)
2. load the death data (from the OMOP database's PERSON table)
3. conduct Kaplan-Meier survival analysis

```
kmf.fit( T_this_pheno, C_this_pheno, label=l_unique_groups[i], alpha=1 )
kmf.plot( ax=ax , xlim=(0,15))
plt.xticks(np.arange(0, 15, 1.0), fontsize=12)
plt.yticks( fontsize=12)
#plt.legend(loc=3, fontsize=8)
plt.legend(bbox_to_anchor=(.35, .4
            ), fontsize=10)

plt.xlabel('follow-up time (years)' , fontsize=18)
plt.title('')
#plt.title("Survival times of phenotypes after first imaging procedure")
```

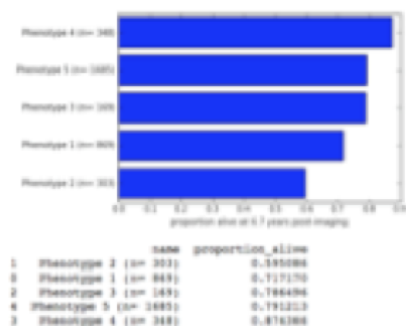
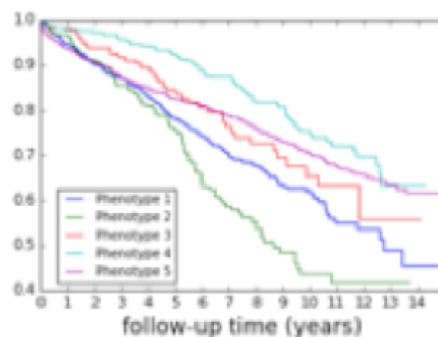


Figure 4.8: Screenshots from the phenotyping pipeline notebook. Shown here are excerpts of code for follow-up analysis that can be conducted on the phenotypes. Kaplan-Meier analysis is shown with Kaplan-Meier curve and differential survival shown in bar chart format.

Database Connector

First, a database connector is required in order to retrieve input data from an OMOP database. The user is responsible for entering filtering criteria in order to indicate which database tables are to be used for feature construction, as well as criteria for selecting patients to include in the cohort (Figure 4.6).

Data Transformation

In the data transformation step, data for various concept tables are pulled from the database and transformed into the required data structures for phenotype extraction. For example, if non-negative tensor factorization is used as the algorithm, then events from separate concept tables (e.g., drug exposures and condition occurrence) can be aligned with respect to time and co-occurrences on a per patient basis can be extracted and represented in tensors.

Phenotype Extraction

The phenotyping algorithm is run in the phenotype extraction step. Afterwards, the results are stored in a JSON object, in the results directory. In our example pipeline, we have implemented non-negative tensor factorization for phenotype extraction. The specific steps include construction of tensors based on co-occurrences of events, followed by the iterative non-negative tensor factorization process. Non-negative tensor factorization performs a parts-based decomposition of the tensor, resulting in factors that represent collections of concepts that are relevant to distinct clusters of patients. The Marble [4] implementation of non-negative tensor factorization based phenotyping is implemented in our example notebook pipeline (Figure 4.7). We implemented Marble because it is the most robust and interpretable phenotyping algorithm with an existing open-source Python implementation.

Follow-up Analysis

Following the phenotype extraction step, follow-up analyses of the extracted phenotypes can be conducted by the user. In our example notebook, we have implemented a survival analysis procedure which performs a Kaplan-Meier analysis on the discovered phenotypes as they relate to survival after a specified index date (Figure 4.8). In order to conduct the analysis, dates of death should be present as an OMOP demographic concept in the database.

Portable Results

The results of the phenotyping discovery pipeline are exported into a common results directory. Two JSON objects that describe the discovered phenotypes are exported. The first JSON object contains the phenotype definitions with any patient information, details about the algorithm, as well as details of each phenotype discovered. The second JSON object contains the patient phenotype assignments. The phenotype definition JSON object is shown in Figure 4.9, and the phenotype assignment JSON object is displayed in Figure 4.10.

Use Cases

Using the saved phenotype discovery pipeline, the user has the capability to construct phenotypes on any new set of data (Case Study 1, Figure 4.11B). Furthermore, using any set of phenotype definitions discovered from the pipeline, the user has the capability to score patients from a separate dataset on the existing phenotype definitions (Case Study 2, Figure 4.11C). Finally, the user has the capability to take the existing pipeline and customize the phenotype extraction section in order to develop new algorithms for unsupervised phenotyping (Case Study 3, Figure 4.11D).

```

{
  "configuration": {
    "algorithm": {
      "algorithm_description": "Non-negative tensor factorization (Marble)",
      "algorithm_parameters": {
        "alpha": 1,
        "mode_labels": ["PATIENT", "DIAGNOSIS", "MEDICATION"],
        "gamma": [0.001, 0.06, 0.04]
      }
    }
  },
  "tables": [
    { "table_name": "CONDITION_OCCURRENCE" },
    { "table_name": "DRUG_EXPOSURE" }
  ],
  "phenotype_descriptions": [
    {
      "phenotype_name": "Phenotype 4",
      "features": {
        "CONDITION_OCCURRENCE": [
          {
            "description": "Atrial fibrillation (HCC)",
            "weight": 0.16,
            "concept_id": 727,
            "condition_source_concept_id": "427.3",
            "vocabulary_id": 2,
            "vocabulary_name": "ICD9CM"
          },
          {
            "description": "Coronary atherosclerosis of unspecified type of vessel, native or graft",
            "weight": 0.22,
            "concept_id": 690,
            "condition_source_concept_id": "414.00",
            "vocabulary_id": 2,
            "vocabulary_name": "ICD9CM"
          }
        ],
        "DRUG_EXPOSURE": [
          {
            "description": "FUROSEMIDE 20 MG PO TABS",
            "weight": 0.38,
            "concept_id": 77,
            "drug_concept_id": "1742",
            "vocabulary_id": 10,
            "vocabulary_name": "GPI"
          },
          {
            "description": "NITROSTAT 0.4 MG SL SUBL",
            "weight": 0.35,
            "concept_id": 94,
            "drug_concept_id": "1299",
            "vocabulary_id": 10,
            "vocabulary_name": "GPI"
          }
        ]
      }
    }
  ]
}

```

Phenotyping algorithm description and configuration parameters

Concept tables used in phenotyping

Phenotype Definitions

For each concept in phenotype definition: concept ID, concept name, relative weight, vocabulary name, vocabulary ID

Figure 4.9: An illustration of the phenotype definitions generated from the pipeline, which are saved in a portable JSON object. Data about the particular phenotyping algorithm and configurations (e.g., parameter tuning) used, OMOP tables used, associated concept vocabularies used, and detailed descriptions of each phenotype are included. In the descriptions of each phenotype, each particular concept included in the phenotype definition is included, along with the associated concept vocabulary and code, as well as the associated weight of contribution of each concept to the final phenotype.

```

{
  "configuration": {
    "algorithm": {
      "algorithm_description": "Non-negative tensor factorization (Marble)",
      "algorithm_parameters": {
        "alpha": 1,
        "mode_labels": ["PATIENT", "DIAGNOSIS", "MEDICATION"],
        "gamma": [0.001, 0.06, 0.04]
      }
    },
    "cohort": {
      "name": "Sutter Health Heart Failure Dataset",
      "database": "/SQL/database/location:"
    }
  },
  "tables": [
    { "table_name": "CONDITION_OCCURRENCE" },
    { "table_name": "DRUG_EXPOSURE" }
  ],
  "phenotype_cohorts": [
    {
      "phenotype_name": "Phenotype 1",
      "patients": [13, 21, 30, 45, 64, 66, 87, 111, 132, 137],
    },
    {
      "phenotype_name": "Phenotype 2",
      "patients": [613, 231, 70, 22, 66, 77, 58, 62],
    },
    {
      "phenotype_name": "Phenotype 3",
      "patients": [69, 7, 3, 27, 92, 888, 37, 444, 44, 27],
    }
  ]
}

```

Phenotyping algorithm details

Cohort description and OMOP database details

OMOP database tables used in phenotyping

Phenotype cohort details (patient assignments for each phenotype)

Figure 4.10: An illustration of the phenotype cohorts generated from the pipeline, which are saved in a portable JSON object.

Results

Data

In this section we describe the two datasets used in each of the 3 case studies.

Sutter Health: The source population for this study was primary care patients from Sutter Palo Alto Medical Foundation (PAMF) Clinics, multispecialty group practices with large primary care practices. Sutter PAMF includes 497 primary care providers in northern California Bay Area and coastal areas that provide care to 700,000 patients. **CMS Linkable 2008-2010 Medicare Data Entrepreneurs Synthetic Public Use File (DE-SynPUF):** The CMS Linkable 2008-2010 Medicare Data Entrepreneurs Synthetic Public Use File (DE-SynPUF) is a publicly available dataset that comprises a set of 2.1 million patients. From this dataset, we identified a set of 3,983 patients who met the inclusion criteria for heart failure status according to the Sutter Health dataset.

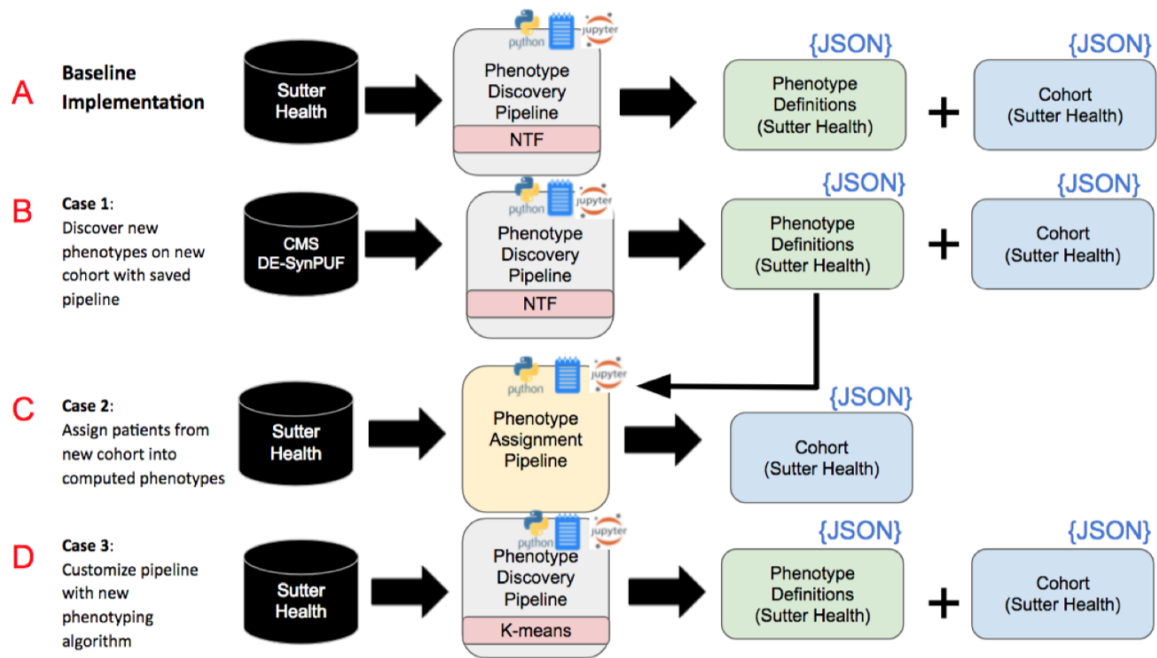


Figure 4.11: An illustration of possible workflows in different use cases of the phenotyping pipeline. The specific data sources used in our use case implementations are shown in the figure. (A) in the baseline case, the phenotype discovery pipeline is used to generate phenotype definitions and cohorts (patients belonging to each phenotype), using the non-negative tensor factorization (NTF) algorithm. (B) In Case 1, a pipeline can be used to discover phenotypes from a cohort, and a notebook and saved configurations can be saved. Subsequently, a new pipeline can be launched with the saved notebook and configurations, using a different cohort. (C) In Case 2, a phenotype assignment pipeline can be launched which takes the phenotype definitions from a previous iteration of the phenotype discovery pipeline as input, along with a new dataset, and outputs assignments of patients from the new dataset into the phenotype definitions loaded. (D) In Case 3, the phenotype discovery pipeline is customized to the users preferences, which may involve swapping out a different phenotyping algorithm; in this example, K-means is used as the phenotype discovery algorithm. Note that different phenotyping algorithms and datasets can be used depending on the specific scenario.

Determination of Heart Failure Status

A rule-based algorithm based on inclusion criteria of heart failure diagnosis (ICD-9) codes from the EHR was developed and implemented on the Sutter Health dataset. Note that EpicCare EHR was installed at Sutter-PAMF in 1999. Assignment of patients to a PCP is documented in an EHR structured field. Incident HF cases were identified from Sutter-PAMF between January 1, 2001 and December 31, 2010 and from Sutter PAMF from May 16, 2000 to May 23, 2013.

Criteria for incident onset of HF were adopted from [Gurwitz et al., 2013] [38] while criteria for HF subtypes were defined in detail elsewhere [45, 38, 46], but relied on qualifying ICD-9 codes for HF with a minimum of three clinical encounters occurring within 12 months of each other. Qualifying cases must be assigned to a PCP in the appropriate EHR structured field. Qualifying ICD-9 codes included 398.91, 402.01, 402.11, 402.91, 404.01, 404.03, 404.11, 404.13, 404.91, 404.93, 428.0, 428.1, 428.20, 428.21, 428.22, 428.23, 428.30, 428.31, 428.32, 428.33, 428.40, 428.41, 428.42, 428.43, 428.9, EP427, EP428, EP429, EP431, EP432, and EP433. The date of diagnosis (HFDx) was assigned as follows. If the time between the first and second HF diagnosis codes is less than or equal to 365 days, and the time between the first and third HF diagnosis codes was less than or equal to 547 days, then select the first HF diagnosis code as the HFDx. Otherwise, select the second HF diagnosis code as the HF diagnosis date (HFDx). Analysis was limited to patients who were 40 to 84 at the time of HF diagnosis. A total of 4,370 incident HF cases were identified from Sutter-PAMF over the time period from 2000 to 2013.

An illustration of the data filtering to include only HF cases is shown in Figure 4.12.

Case Study 1

For this case study, we first used the pipeline to discover phenotypes for heart failure in the Sutter Health dataset (Figure 4.7).

Data processing: The features pertaining to conditions were extracted from both of the

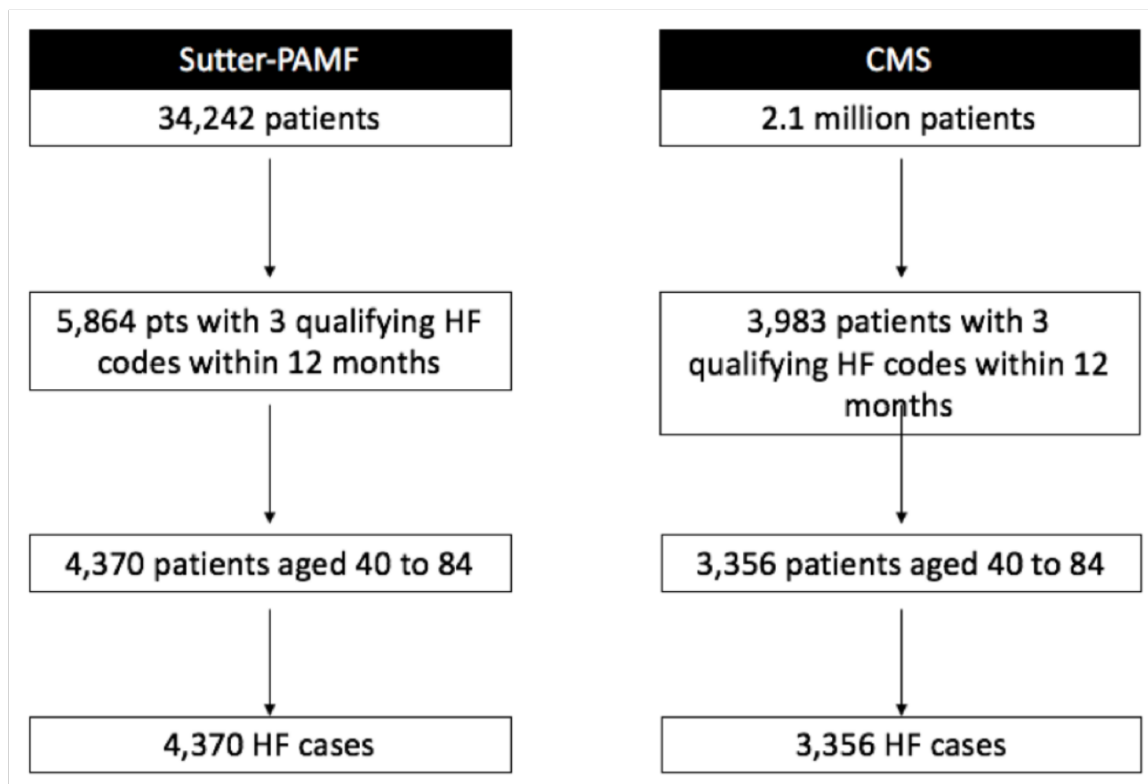


Figure 4.12: An illustration of the data filtering required for cohort construction. Patients from the Sutter Health dataset who met inclusion criteria for heart failure case definition were identified. Corresponding patients from the CMS DE-SynPUF dataset who met the same criteria were subsequently identified.

heart failure cohorts. From each of the cohorts, conditions were coded in the form of ICD-9 codes. Within each cohort, the conditions were sorted by order of frequency of occurrence among all events, and the 500 most frequently occurring conditions were kept for analysis while all others were filtered out.

Tensor Construction: Construction of tensors was required for implementation of the non-negative tensor factorization method for phenotyping. Let the observation window for any given patient be defined as a two-year period of time before the heart failure diagnosis date. That is, if HFDx is the heart failure diagnosis date, the observation window ranges from day HFDx - 730 up until HFDx. A 2nd order tensor was constructed for each of the two cohorts, whereby the first mode of the tensor represented patients and the second mode of the tensor represented conditions. If a condition occurred in the observation window, the appropriate entry of the tensor is incremented. Thus the resulting tensor was a count tensor of all events. The tensor was converted to a binary tensor for the phenotyping discovery analysis.

Phenotype Discovery via Non-negative Tensor Factorization: We used the tensor factorization algorithm Marble [4], to decompose the constructed tensor into components. Marble decomposes the tensor into components that represent subtypes of patients in the original tensor. The Marble tensor factorization problem is formulated as the following:

$$V = [\lambda; \dots A^{(1)}, \dots; A^{(N)}]$$

where V represents the original tensor, A represent factors and N represents the number of dimensions of the tensor. Each A(n) is a linear combination of R different components, where R represents the number of desired subgroups we wish to obtain. The following constraints are imposed:

$$\lambda_n \in [0, +\infty)$$

imposes constraint on non-negativity indicating that the resultant factors are composed of non-negative values (i.e., 0 or greater),

$$A^{(n)} \in \{0, [\gamma_n, 1]\}^{I_n \times 1}$$

imposes sparsity where can be tuned to obtain the desired proportion of non-zero entries per dimension,

$$\|A^n\|_n = 1$$

imposes a stochastic constraint whereby all weights along a given dimension per factor sum to 1, which allows for easy interpretation of relative contributions of features to the final subtype. The end result is a set of R phenotypes each of which are represented as an outer product of 2 vectors. Each of the 2 vectors (which are factors of the phenotype) contains a sparse set of non-negative elements while the rest of the elements are 0. The non-negative elements are elements that belong to the subgroup. For each of the R phenotypes, each factors weighted elements sum to 1, and each non-zero elements weight represents that elements relative contribution to that phenotype definition.

The specific parameters we used were:

$$\alpha = 100, \gamma = [0.001, 0.02], R = 10$$

for the Sutter Health dataset and

$$\alpha = 100, \gamma = [0.001, 0.06], R = 10$$

for the CMS DE-SynPUF dataset.

Saved Results and Configurations: After conclusion of the phenotype discovery process, we ported the saved configurations from the pipeline to discover phenotypes for heart

failure in the CMS DE-SynPUF dataset. Examples of phenotypes obtained from the Sutter Health dataset are shown in Figure 4.13. Examples of phenotypes obtained from the CMS DE-SynPUF dataset are shown in Figure 4.14. Note that while we extracted 10 phenotypes for each of the datasets, we have only shown two examples from each dataset. Further re-search studies should focus on identifying and refining robust clinical phenotypes for heart failure.

Phenotype 1		Phenotype 2	
Number of Patients: 350 (9% of cohort)	Weight	Number of Patients: 284 (7% of cohort)	Weight
Cough	0.32	Type II or unspecified type diabetes mellitus	0.22
Chronic airway obstruction	0.23	Other and unspecified hyperlipidemia	0.21
Unspecified essential hypertension	0.17	Unspecified essential hypertension	0.17
Unspecified asthma	0.15	Esophageal reflux	0.15
Other and unspecified hyperlipidemia	0.12	Essential hypertension, benign	0.15
		Coronary atherosclerosis	0.10

Figure 4.13: An illustration of the heart failure phenotypes found in the Sutter Health dataset. For each phenotype, the number of patients (% of cohort) and relative weight of contribution of each feature are shown.

Case Study 2

For this case study, we use the phenotyping assignment pipeline to assign HF patients from the Sutter Health dataset to the phenotype definitions obtained from the CMS DE-SynPUF dataset (Figure 4.8). The patients from the Sutter Health dataset are projected onto the space of phenotypes extracted from the CMS SynPUF dataset. The results for the phenotype assignments were saved into a JSON object that can be easily ported into other workflows for future follow-up analysis.

Phenotype 7		Phenotype 9	
Number of Patients: 499 (12% of cohort)	Weight	Number of Patients: 423 (11% of cohort)	Weight
Benign essential hypertension	0.30	HYPERTENSION NOS	0.48
HYPERTENSION NOS	0.17	PREMATURE BEATS NOS	0.23
Other and unspecified hyperlipidemia	0.16	Sinoatrial node dysfunction	0.15
Impaired glucose tolerance test (oral)	0.14	Chronic kidney disease, Stage III	0.14
Glycosuria	0.13		
Paroxysmal tachycardia, unspecified	0.11		

Figure 4.14: An illustration of the heart failure phenotypes found in the CMS SynPUF dataset. For each phenotype, the number of patients (% of cohort) and relative weight of contribution of each feature are shown.

Case Study 3

For this case study, we customized the phenotyping pipeline to use a K-means algorithm for determining phenotypes, in place of the original non-negative tensor factorization algorithm (Figure 5D). A total of 10 clusters were computed. In order to determine the features defining each phenotype, a feature selection method via recursive feature elimination was used [94]. The K-means algorithm produces a set of 10 hard clusters, along with assignment of patients into clusters. Each cluster obtained via K-means denotes membership of a patient into a cluster. In order to identify features (conditions) that define each cluster, feature selection via recursive feature elimination [98] is employed in order to choose the most likely features that are most predictive for the patient assignment into a particular cluster. For this particular implementation, a support vector regression estimator is trained on the initial raw feature set (feature matrix of patient conditions from the CONDITION_OCCURRENCE table in the database), weights are assigned to each feature, and the feature set is recursively pruned until a desired number of features (6 features) is eventually

reached for each phenotype.

Discussion

We proposed and implemented a scalable, modular pipeline for performing phenotyping tasks on patient data. We applied this pipeline to 3 separate case studies that demonstrate the portability and customizability of our pipeline.

Standardization of Workflows

With the increasing tsunami of healthcare data being produced and stored, and the subsequent rise in interest in unsupervised phenotyping algorithms for patient segmentation and disease subtype discovery, it will be crucial for researchers to adopt standardized phenotyping strategies, frameworks and workflows. Standardization of practices allows for ease of collaboration, validation and reproducibility of results in the healthcare analytics practices involving electronic health record data [99, 100].

Our innovation is the first such implementation of a data science analytics environment for performing phenotyping tasks on datasets that may be obtained in the OMOP common data model. Our work should motivate future initiatives on releasing share-able, actionable code samples that can be easily modified and tuned to any medical data scientists preferences.

It is important to note that our study was intended to showcase the potential of an open-source, scalable and modular phenotyping framework, using the lab notebook form factor with which many medical data scientists are familiar.

Facilitating Phenotype Validation

Validation of phenotypes obtained is critical, and will only be increasing in interest and importance with the rise in unlabeled medical data and increasing usage of feature extraction strategies such as natural language processing and deep learning as applied to data sources

like clinical notes, genomic data, behavioral data and electronic health records. Although we showcased the usage of our phenotyping pipeline notebooks for subtyping of heart failure patients, our study was not intended to be a robust discovery and validation process for heart failure phenotypes. Future research should leverage standardized phenotyping analytic pipelines such as ours, in an effort to iteratively identify and validate phenotypes across various patient cohorts of interest.

Facilitating Subtype Discovery for Diseases

It is readily apparent in many instances in clinical care today that a one size fits all approach to patient management is flawed and leads to suboptimal outcomes. For example, it has been shown that patients with certain genetic biomarkers respond differently to standard therapies for conditions including cancer. In the area of lung cancer treatment, patients with non-small cell lung cancer that have EGFR mutations have traditionally been treated with EGFR tyrosine kinase inhibitors but may not respond to other therapies [101].

Heart failure is another disease that exhibits high complexity and heterogeneity among patients, making it difficult to treat. Particularly in cases of heart failure with preserved ejection fraction (HFpEF), in which patients have ejection fractions greater than 50%, patient management is largely difficult because patients typically have multiple comorbidities and also because highly efficacious therapies that work on a broad range of patients have not been developed [39]. Unsupervised phenotyping, including with non-negative tensor factorization approaches similar to that which we implemented in our case study (Case 1), has been proposed widely for purposes such as discovering subtypes of the disease [36, 40, 34].

Future studies should leverage standardized phenotyping analytic pipelines, such as ours, in an effort to iteratively identify and validate phenotype definitions for disease subtypes. Employing the use cases outlined in our results, multiple institutions may wish to extract phenotype definitions in distinct cohorts and compare them (Case 1), to assign a

new set of patients' phenotype definitions previously discovered in different cohorts (Case 2), or to build, debug and deploy customized phenotyping protocols involving fine-tuned, customized phenotyping algorithms (Case 3).

Conclusion

We proposed and implemented a modular and executable data science pipeline for performing phenotype discovery tasks via an easily shareable notebook environment. We applied this pipeline successfully to several case studies, all of which illustrated the portability and customizability of the pipeline.

CHAPTER 5

DISCUSSION, CONCLUSIONS AND AND CLOSING REMARKS

We have proposed, implemented, and validated multiple algorithms, tools and applications that help to tackle existing chronic disease management problems. Discussions about specific projects and future academic research work have been addressed in the preceding chapters. In this final chapter we discuss the practical, real-world impacts that our research may bring about.

5.1 Overall Discussion and Conclusions

This thesis focused on developing and implementing prototype versions of algorithms and software for automatically segmenting patients and predicting future risks of disease. Most of our work focused on technical implementation and performance analysis typical for evaluating such technologies at the research stage. The logical next step is to translate these technologies into commercial products and services in order to derive real world clinical and financial impact.

5.1.1 Creating and Capturing Value in the Healthcare Analytics Space with Phenotyping Technologies

Financial Impact Analysis

As we push for greater adoption and commercialization of phenotyping technology, it is important to note the potential financial impacts that phenotyping technologies can provide to major stakeholders of the healthcare system.

Pharmaceutical Companies

It is widely known that there are various inefficiencies present in the traditional pharmaceutical research and development value chain.

The area of clinical trial management is perhaps one of the areas of the pharmaceutical research and development value chain most ripe for disruption. Take the example of a recent clinical trial for dementia, which was conducted by Axovant [102, 103]. The trial had failed to show a positive effect of intepirdine on performance in two cognitive tasks in Alzheimers patients: Alzheimer’s Disease Assessment Scale-Cognitive Subscale (ADAS-Cog) and Alzheimer’s Disease Cooperative Study-Activities of Daily Living scale (ADCS-ADL). Unfortunately the future of intepirdine remains uncertain. However, if a phenotyping study was done on the cohort of patients during the trial enrollment phase, a statistical analysis plan involving sub-cohort analysis could have been designed prior to the trial that could have allowed the company to hedge the trial.

The financial impact of such a process is massive. In the case of Axovant, they could have released a new Alzheimer’s drug to the market, even if it was just for a small targeted subpopulation of patients. A new therapy could have potentially garnered the company an additional \$1.5B in top-line revenue per year once the drug hit peak sales. And this could have been accomplished without the need for running a separate trial - the sub-cohort analysis could have been baked into Axovant’s analysis plan.

Providers

Providers can potentially derive immense value from phenotyping algorithms, systems or applications. Due to impending regulatory changes within the next 5-10 years, providers will become increasingly more conscious about controlling the cost of care and improving long term outcomes. The macro trend of increasing managed care organizations can be seen from historical data from the past decade (Figure 5.1).



Figure 5.1: There has been a rise in the number of accountable care organizations in the United States for the past 6 years. This trend is likely to continue in the future amid a shifting regulatory environment as the United States healthcare system shifts from a service-based reimbursement model an outcome-based and cost-based reimbursement model.

Health IT companies

One potentially valuable use case of phenotyping algorithms, systems and applications is in digital therapeutics. There are many use cases revolving around digital therapeutics in the form of mobile applications to help patients with preventative care. For example, the company Omada Health (<https://www.omadahealth.com/>) provides a mobile app solution that monitors patients' weight loss over time. Weight loss serves as a strong preventative measure that is linked to lower risk of diabetes, and thus remains a strong interest to diabetes patients and their care providers. A potential roadblock to preventative management is that certain subgroups of patients are potentially less likely to lose weight given a specific diet during a specified time relative to other subgroups of patients. It would be immensely valuable to companies such as Omada Health to have a way to identify these patient cohorts ahead of time in order to provide personalized therapy to them. For example, more rigorous patient engagement strategies may be used for the more high-risk patient subgroups that may not be able to achieve weight loss goals as easily as others. Furthermore, the company may also be able to charge different patient cohorts differently based on differences

in personalized digital therapeutic therapy required to achieve desired results.

Payers

Perhaps one of the most obvious use cases of phenotyping is in the payer side (i.e., insurance companies). Payers can extract immediate financial impact from algorithms, tools or applications that segment or stratify patients into risk pools with varying degrees of severity or prognosis.

One experimental strategy that can be used to demonstrate this particular financial impact is to perform phenotype discovery on a cohort using electronic health record (EHR) data from a health system. The phenotype discovery strategy would utilize diagnosis and medication data from the patients, in a similar fashion as demonstrated in the experiments performed around the *Rubik* algorithm. After phenotypes are discovered, the utilization of resources within each patient phenotype can be tracked. To do so, the procedures documented for each patient within a phenotype would be extracted. The expected cost per procedure can be obtained via the Medicare Provider Utilization and Payment Data database[104]. After average utilization cost is computed within each phenotype, the phenotypes can be compared with respect to expected expenditures for each phenotype. A payer can utilize such insights to determine which of its beneficiaries fit the profile for a specific phenotype, and deliver differentiated product offerings (i.e., charge varying amounts for different patient phenotype profiles).

Outcome Analysis

Apart from financial impact, the main other way novel phenotyping technologies presented in this thesis deliver value to healthcare stakeholders is in improvement of patient outcomes.

For pharmaceutical companies and for payers, a large area of interest lies in real-world evidence, which involves leveraging data and insights from outside of the clinical trial setting to further the understanding of the impact of drugs being tested.

One prominent example of this is survival analysis. Survival analysis entails measuring the long term survival of subgroups of patients. In a clinical trial setting, survival over time are typically measured across different treatment arms. Figure 5.2 shows an example of a Kaplan-Meier plot from a prototypical analysis.

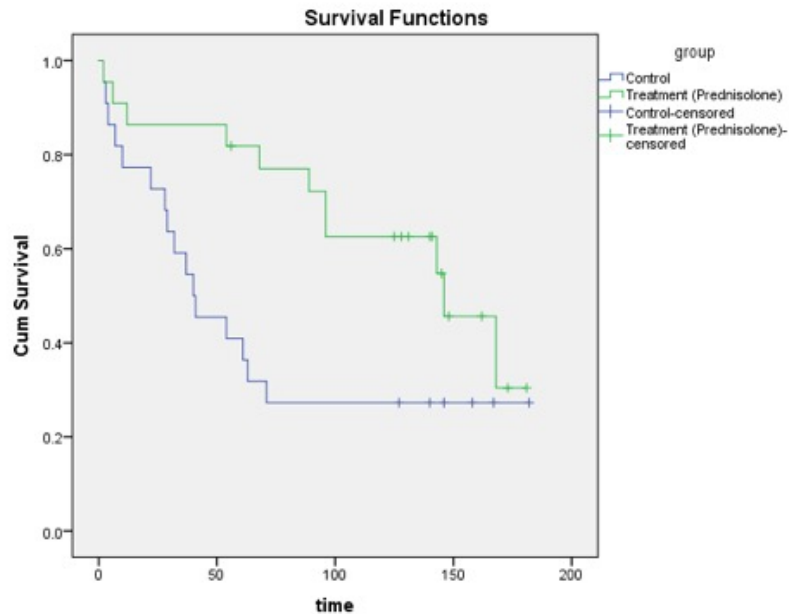


Figure 5.2: An example of a Kaplan-Meier plot. In this particular implementation, two subgroups of patients were tracked over time with respect to survival rates. Phenotyping algorithms may be validated with survival analysis, as they may potentially identify patient subgroups that exhibit differential response to therapy.

One real-world use case of phenotyping algorithms such as Rubik is in subset analysis of the treatment arms. For example, in the example in Figure 5.2, it is possible that there is a subset of patients undergoing prednisone therapy (green line) that did worse than the placebo group, and a subset that did better than the placebo group. If a phenotyping analysis was done in order to pick clinical cohorts, this relationship could have been uncovered. A finding like this is potentially instrumental in helping to refine follow-up trials for pharmaceutical companies, or to allow pharmaceutical companies to hedge their trials by designing more robust data analysis plans before commencing the trial. More rigorous analysis in this case could be instrumental in identifying patient cohorts that respond best

to a particular therapy, and this insight could be further leveraged by providers in order to deliver targeted therapies for substantially improved patient outcomes.

5.1.2 Validation of Phenotyping Algorithms, Tools and Applications

Continued validation of the technologies presented in this thesis will be beneficial for developing further understanding of unmet clinical needs in the healthcare analytics space and for efforts to commercialize technologies based on our research findings.

To validate financial impact of phenotyping algorithms such as Rubik, targeted studies should be done to analyze potential costs of patient phenotypes discovered with the algorithm. Future work should focus on analyzing procedures typically received by patients belonging in a phenotype and comparing overall cost differences between phenotypes. To do this validation, claims datasets such as from the Center for Medicare and Medicaid Services can be leveraged. A schematic of the analysis plan is outlined in Figure 5.3.

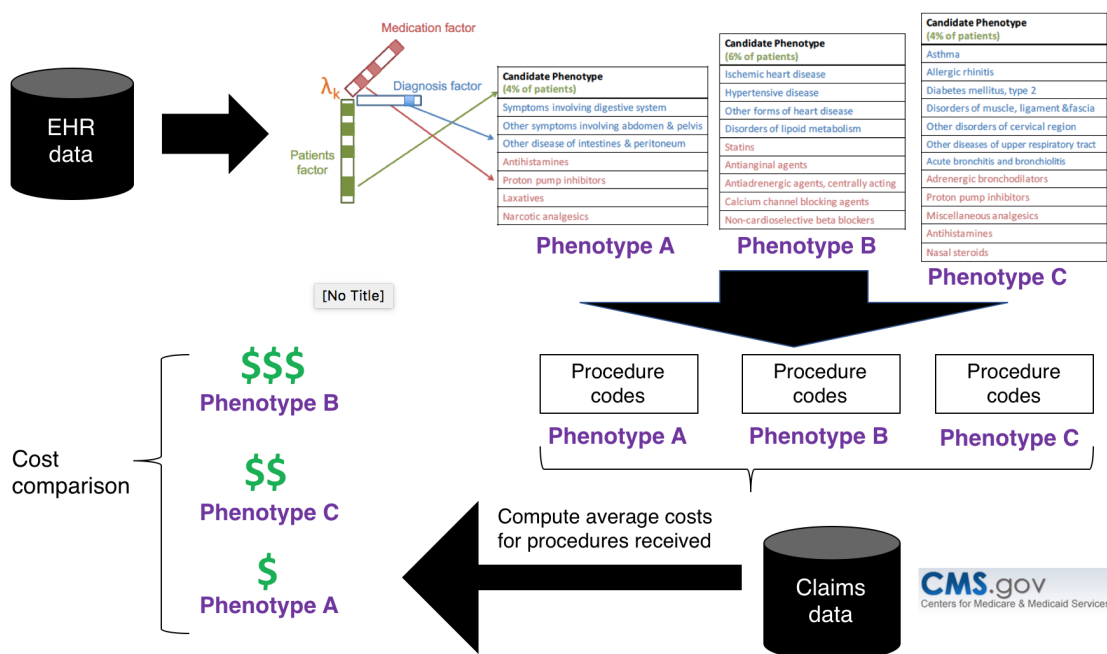


Figure 5.3: Sample analysis plan for a cost comparison analysis of phenotypes derived from a phenotyping algorithm.

Once the cost differences are identified, the insight could be used by both payers and

providers to help them with their goals. The final step in validating the financial impact is to conduct a pilot study with the phenotyping technology in a health system or insurance company. Two study arms would be incorporated in the study: one where the health system or insurance company uses the phenotyping technology to uncover phenotypes and stratify patients by potential cost; and one where the health system or insurance company does not use the technology. Separate patients would be included in each arm. In both arms the pilot would last for a period of 2 fiscal quarters. The financial impact would be measured during these two quarters. For an insurance company, the difference in premiums collected as revenues would be measured for statistical significance. For a health system, the difference in total spending would be measured for statistical significance. The demonstration of financial impact through this validation process is instrumental for marketing and selling purposes.

To further validate phenotyping algorithms, outcome analysis can also be performed. One way to do so is to perform either retrospective or prospective studies that provide insight on the differences in medical outcomes between different phenotypes. One way to perform a retrospective study is to perform phenotype discovery on a set of patients from a health system, using data from a specific observation window. For example, in a cardiology clinic, the observation window may be defined as a two-year period immediately preceding the date before the first date of cardiac imaging (cardiac MRI or echocardiogram). After phenotypes are discovered, the long-term survival of patients within each phenotype can be tracked and differentiated. A phenotyping algorithm that can deliver high insightful value should be able to identify patient segments with high risk profiles, and this would be seen in the analysis as patients in a phenotype who have a lower survival rate compared to patients in other phenotypes after a set amount of time after the observation window. An example can be seen in Figure 5.4. In this example, a segment of patients who exhibited multiple co-morbidities including diabetes, kidney disease and sepsis, exhibited a 40% survival rate at a 15-year time point post initial cardiac imaging while less high-risk segments of patients

such as a segment who only had obesity as a condition, exhibited over 90% survival rate at the 15 year time point.

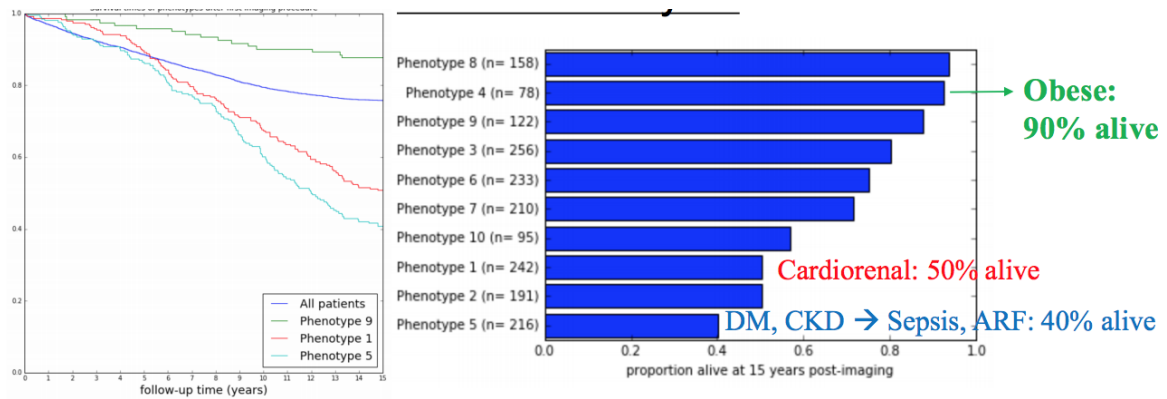


Figure 5.4: Cardiovascular patients undergoing cardiac imaging (cardiac MRI, echocardiogram) were automatically grouped into 10 phenotypes, which show vast differences in survival after over a decade of follow-up.

5.2 Closing Remarks

We have demonstrated several key innovations pertaining to algorithms, tools and applications across all three major segments of a typical medical data science value chain: phenotype discovery, predictive modeling, and model deployment. We have proposed future work including segmentation of heart failure patients, predictive modeling via deep learning algorithms for early detection of heart failure, and model deployment of a recurrent neural network based algorithm for early detection of heart failure in a large-scale prospective biomarker study. There are various avenues for future research

REFERENCES

- [1] J. C. Denny *et al.*, “Systematic comparison of phenome-wide association study of electronic medical record data and genome-wide association study data,” *Nature Biotechnology*, vol. 31, no. 12, pp. 1102–1111, 2013.
- [2] J Pathak, A. N. Kho, and J. Denny, “Electronic health records-driven phenotyping: Challenges, recent advances, and perspectives,” *Journal of the American Medical Informatics Association*, vol. 20, no. e2, e206–e211, 2013.
- [3] A. N. Kho *et al.*, “Electronic medical records for genetic research: Results of the emerge consortium,” *Science Translational Medicine*, vol. 3, no. 79, 79re1, 2011.
- [4] J. C. Ho, J. Ghosh, and J. Sun, “Marble: High-throughput phenotyping from electronic health records via sparse nonnegative tensor factorization,” in *KDD*, 2014.
- [5] J. Liu, P. Musialski, P. Wonka, and J. Ye, “Tensor completion for estimating missing values in visual data,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 1, pp. 208–220, 2013.
- [6] E. Acar, T. G. Kolda, and D. M. Dunlavy, “All-at-once optimization for coupled matrix and tensor factorizations,” *ArXiv preprint arXiv:1105.3422*, 2011.
- [7] J. Kim and H. Park, “Fast nonnegative tensor factorization with an active-set-like method,” in *High-Performance Scientific Computing*, Springer, 2012, pp. 311–326.
- [8] E. Acar, D. M. Dunlavy, T. G. Kolda, and M. Mørup, “Scalable tensor factorizations for incomplete data,” *Chemometrics and Intelligent Laboratory Systems*, vol. 106, no. 1, pp. 41–56, 2011.
- [9] I. Davidson, S. Gilpin, O. Carmichael, and P. Walker, “Network discovery via constrained tensor analysis of fmri data,” in *KDD*, 2013.
- [10] G. Hripcsak and D. J. Albers, “Next-generation phenotyping of electronic health records,” *Journal of the American Medical Informatics Association*, 2012.
- [11] Y. Chen *et al.*, “Applying active learning to high-throughput phenotyping algorithms for electronic health records data,” *JAMIA*, 2013.
- [12] J. C. Ho, J. Ghosh, S. R. Steinhubl, W. F. Stewart, J. C. Denny, B. A. Malin, and J. Sun, “Limestone: High-throughput candidate phenotype generation via tensor factorization,” *Journal of biomedical informatics*, vol. 52, pp. 199–211, 2014.

- [13] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, “Distributed optimization and statistical learning via the alternating direction method of multipliers,” *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.
- [14] Z. Lin, R. Liu, and Z. Su, “Linearized alternating direction method with adaptive penalty for low-rank representation,” in *NIPS*, 2011.
- [15] A. Smilde, R. Bro, and P. Geladi, *Multi-way analysis: Applications in the chemical sciences*. John Wiley & Sons, 2005.
- [16] J. C. Denny *et al.*, “Phewas: Demonstrating the feasibility of a phenome-wide scan to discover gene disease associations,” *Bioinformatics*, vol. 26, no. 9, pp. 1205–1210, 2010.
- [17] Center for Medicare and Medicaid Services. (). CMS 2008-2010 data entrepreneurs synthetic public use file (DE-SynPUF), (visited on 2015).
- [18] E. C. Chi and T. G. Kolda, “On tensors, sparsity, and nonnegative factorizations,” *SIAM Journal on Matrix Analysis and Applications*, vol. 33, no. 4, pp. 1272–1299, 2012.
- [19] C. for Disease Control and P. (CDC), “Chronic diseases at a glance 2009,” *Technical report*, CDC, 2009.
- [20] T. G. Kolda and B. W. Bader, “Tensor decompositions and applications,” *SIAM review*, vol. 51, no. 3, pp. 455–500, 2009.
- [21] M. Welling and M. Weber, “Positive tensor factorization,” *Pattern Recognition Letters*, vol. 22, no. 12, pp. 1255–1261, 2001.
- [22] J. D. Carroll, S. Pruzansky, and J. B. Kruskal, “Candelinc: A general approach to multidimensional analysis of many-way arrays with linear constraints on parameters,” *Psychometrika*, vol. 45, no. 1, pp. 3–24, 1980.
- [23] A. Narita, K. Hayashi, R. Tomioka, and H. Kashima, “Tensor factorization using auxiliary information,” *Data Mining and Knowledge Discovery*, vol. 25, no. 2, pp. 298–324, 2012.
- [24] B. Ermiş, E. Acar, and A. T. Cemgil, “Link prediction in heterogeneous data via generalized coupled tensor factorization,” *Data Mining and Knowledge Discovery*, vol. 29, no. 1, pp. 203–236, 2015.
- [25] A. Beutel, A. Kumar, E. E. Papalexakis, P. P. Talukdar, C. Faloutsos, and E. P. Xing, “Flexifact: Scalable flexible factorization of coupled tensors on hadoop,” in *SDM*, 2014.

- [26] E. E. Papalexakis, T. M. Mitchell, N. D. Sidiropoulos, C. Faloutsos, P. P. Talukdar, and B. Murphy, “Turbo-smt: Accelerating coupled sparse matrix-tensor factorizations by 200x,” in *SDM*, 2014.
- [27] M. Signoretto, Q. T. Dinh, L. De Lathauwer, and J. A. Suykens, “Learning with tensors: A framework based on convex optimization and spectral regularization,” *Machine Learning*, vol. 94, no. 3, pp. 303–351, 2014.
- [28] R. Tomioka and T. Suzuki, “Convex tensor decomposition via structured Schatten norm regularization,” in *NIPS*, 2013.
- [29] Y. Liu, F. Shang, H. Cheng, J. Cheng, and H. Tong, “Factor matrix trace norm minimization for low-rank tensor completion,” in *SDM*, 2014.
- [30] D. Kressner, M. Steinlechner, and B. Vandereycken, “Low-rank tensor completion by Riemannian optimization,” *BIT Numerical Mathematics*, vol. 54, no. 2, pp. 447–468, 2014.
- [31] C. Mu, B. Huang, J. Wright, and D. Goldfarb, “Square deal: Lower bounds and improved relaxations for tensor recovery,” in *ICML*, 2013.
- [32] B. Romera-Paredes and M. Pontil, “A new convex relaxation for tensor completion,” in *NIPS*, 2013.
- [33] Y. Xu and W. Yin, “A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion,” *SIAM Journal on Imaging Sciences*, vol. 6, no. 3, pp. 1758–1789, 2013.
- [34] S. J. Shah, D. W. Kitzman, B. A. Borlaug, L. Van Heerebeek, M. R. Zile, D. A. Kass, and W. J. Paulus, “Phenotype-specific treatment of heart failure with preserved ejection fraction: A multiorgan roadmap,” *Circulation*, vol. 134, no. 1, pp. 73–90, 2016.
- [35] L. Li, W.-Y. Cheng, B. S. Glicksberg, O. Gottesman, R. Tamler, R. Chen, E. P. Bottinger, and J. T. Dudley, “Identification of type 2 diabetes subgroups through topological analysis of patient similarity,” *Science translational medicine*, vol. 7, no. 311, 311ra174–311ra174, 2015.
- [36] S. J. Shah, “Precision medicine for heart failure with preserved ejection fraction: An overview,” *Journal of cardiovascular translational research*, vol. 10, no. 3, pp. 233–244, 2017.
- [37] Y. Wang, R. Chen, J. Ghosh, J. C. Denny, A. Kho, Y. Chen, B. A. Malin, and J. Sun, “Rubik: Knowledge guided tensor factorization and completion for health

data analytics,” in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, 2015, pp. 1265–1274.

- [38] J. H. Gurwitz, D. J. Magid, D. H. Smith, R. J. Goldberg, D. D. McManus, L. A. Allen, J. S. Saczynski, M. L. Thorp, G. Hsu, S. H. Sung, *et al.*, “Contemporary prevalence and correlates of incident heart failure with preserved ejection fraction,” *The American journal of medicine*, vol. 126, no. 5, pp. 393–400, 2013.
- [39] S. J. Shah, D. H. Katz, S. Selvaraj, M. A. Burke, C. W. Yancy, M. Gheorghiade, R. O. Bonow, C.-C. Huang, and R. C. Deo, “Phenomapping for novel classification of heart failure with preserved ejection fraction,” *Circulation*, CIRCULATION-AHA-114, 2014.
- [40] D. H. Katz, L. Beussink, A. J. Sauer, B. H. Freed, M. A. Burke, and S. J. Shah, “Prevalence, clinical characteristics, and outcomes associated with eccentric versus concentric left ventricular hypertrophy in heart failure with preserved ejection fraction,” *American Journal of Cardiology*, vol. 112, no. 8, pp. 1158–1164, 2013.
- [41] K. Ng, W. F. Stewart, C. deFilippi, S. Dey, R. J. Byrd, S. R. Steinhubl, Z. Daar, H. Law, A. R. Pressmnan, and J. Hu, “Data-driven modeling of electronic health record data to predict prediagnostic heart failure in primary care,” *Journal of Patient-Centered Research and Reviews*, vol. 3, no. 3, p. 200, 2016.
- [42] K. Ng, S. R. Steinhubl, S. Dey, W. F. Stewart, *et al.*, “Early detection of heart failure using electronic health records: Practical implications for time before diagnosis, data diversity, data quantity, and data density,” *Circulation: Cardiovascular Quality and Outcomes*, vol. 9, no. 6, pp. 649–658, 2016.
- [43] K. Ng, S. Steinhubl, C. deFilippi, S. Dey, and W. Stewart, “Early detection of heart failure using electronic health records: Practical implications for time before diagnosis, data diversity, data quantity, and data density,” *Journal of Patient-Centered Research and Reviews*, vol. 4, no. 3, pp. 174–175, 2017.
- [44] E. Choi, A. Schuetz, W. F. Stewart, and J. Sun, “Using recurrent neural network models for early detection of heart failure onset,” *Journal of the American Medical Informatics Association*, vol. 24, no. 2, pp. 361–370, 2016.
- [45] S. L. Murphy, J. Xu, and K. D. Kochanek, “Deaths: Preliminary data for 2010,” *Natl. Vital Stat. Rep.*, vol. 60, no. 1, pp. 1–52, 2011.
- [46] S. Ather, W. Chan, B. Bozkurt, D. Aguilar, K. Ramasubbu, A. A. Zachariah, X. H. Wehrens, and A. Deswal, “Impact of noncardiac comorbidities on morbidity and mortality in a predominantly male population with heart failure and preserved versus reduced ejection fraction,” *Journal of the American College of Cardiology*, vol. 59, no. 11, pp. 998–1005, 2012.

- [47] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288, 1996.
- [48] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, *et al.*, "Scikit-learn: Machine learning in python," *Journal of machine learning research*, vol. 12, no. Oct, pp. 2825–2830, 2011.
- [49] A. Liaw, M. Wiener, *et al.*, "Classification and regression by randomforest," *R news*, vol. 2, no. 3, pp. 18–22, 2002.
- [50] E. Choi, M. T. Bahadori, J. Sun, J. Kulas, A. Schuetz, and W. Stewart, "Retain: An interpretable predictive model for healthcare using reverse time attention mechanism," in *Advances in Neural Information Processing Systems*, 2016, pp. 3504–3512.
- [51] C.-J. Hsiao, E. Hing, *et al.*, "Use and characteristics of electronic health record systems among office-based physician practices, united states, 2001-2012," 2012.
- [52] J. Pathak, A. N. Kho, and J. C. Denny, "Electronic health records-driven phenotyping: Challenges, recent advances, and perspectives," *Journal of the American Medical Informatics Association*, e206–e211, 2013.
- [53] C. A. McCarty, R. L. Chisholm, C. G. Chute, I. J. Kullo, G. P. Jarvik, E. B. Larson, R. Li, D. R. Masys, M. D. Ritchie, D. M. Roden, *et al.*, "The emerge network: A consortium of biorepositories linked to electronic medical records data for conducting genomic studies," *BMC medical genomics*, vol. 4, no. 1, p. 13, 2011.
- [54] J. Pathak, K. R. Bailey, C. E. Beebe, S. Bethard, D. S. Carrell, P. J. Chen, D. Dligach, C. M. Endle, L. A. Hart, P. J. Haug, *et al.*, "Normalization and standardization of electronic health records for high-throughput phenotyping: The sharpn consortium," *Journal of the American Medical Informatics Association*, vol. 20, no. e2, e341–e348, 2013.
- [55] S. Rea, J. Pathak, G. Savova, T. A. Oniki, L. Westberg, C. E. Beebe, C. Tao, C. G. Parker, P. J. Haug, S. M. Huff, *et al.*, "Building a robust, scalable and standards-driven infrastructure for secondary use of ehr data: The sharpn project," *Journal of biomedical informatics*, vol. 45, no. 4, pp. 763–771, 2012.
- [56] *Fhir specification home page*, <http://www.hl7.org/implement/standards/fhir>, Accessed: 2014-09-25.
- [57] B. M. (2015). Free the health data: Grahame griev on fhir. informationweek healthcare, (visited on 01/25/2015).

- [58] V. University. (2012). Phekb, (visited on 01/25/2015).
- [59] M. Saeed, M. Villarroel, A. T. Reisner, G. Clifford, L.-W. Lehman, G. Moody, T. Heldt, T. H. Kyaw, B. Moody, and R. G. Mark, "Multiparameter intelligent monitoring in intensive care ii (mimic-ii): A public-access intensive care unit database," *Critical care medicine*, vol. 39, no. 5, p. 952, 2011.
- [60] M. L. Braunstein, *Health informatics in the cloud*. New York, NY, USA: Springer-Verlag New York, Inc., 2013.
- [61] T. Bodenheimer and A. Fernandez, "High and rising health care costs. part 4: Can costs be controlled while preserving quality?" *Annals of internal medicine*, vol. 143, no. 1, pp. 26–31, 2005.
- [62] M. Gorelick, P. V. Scribano, M. W. Stevens, T. Schultz, and J. Shults, "Predicting need for hospitalization in acute pediatric asthma," *Pediatric emergency care*, vol. 24, no. 11, pp. 735–744, 2008.
- [63] M. Reznik, S. M. Hailpern, and P. O. Ozuah, "Predictors of early hospital readmission for asthma among inner-city children," *Journal of Asthma*, vol. 43, no. 1, pp. 37–40, 2006.
- [64] C. Feudtner, J. E. Levin, R. Srivastava, D. M. Goodman, A. D. Slonim, V. Sharma, S. S. Shah, S. Pati, C. Fargason, and M. Hall, "How well can hospital readmission be predicted in a cohort of hospitalized children? a retrospective, multicenter study," *Pediatrics*, vol. 123, no. 1, pp. 286–293, 2009.
- [65] J. Salamzadeh, I. Wong, H. Hosker, and H. Chrystyn, "A cox regression analysis of covariates for asthma hospital readmissions," *Journal of Asthma*, vol. 40, no. 6, pp. 645–652, 2003.
- [66] T. Dillon, C. Wu, and E. Chang, "Cloud computing: Issues and challenges," in *Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on*, Ieee, 2010, pp. 27–33.
- [67] A. M.-H. Kuo, "Opportunities and challenges of cloud computing to improve health care services," *Journal of medical Internet research*, vol. 13, no. 3, 2011.
- [68] D. McAullay, G. Williams, J. Chen, H. Jin, H. He, R. Sparks, and C. Kelman, "A delivery framework for health data mining and analytics," in *Proceedings of the Twenty-eighth Australasian conference on Computer Science-Volume 38*, Australian Computer Society, Inc., 2005, pp. 381–387.
- [69] K. Ng, A. Ghoting, S. R. Steinhubl, W. F. Stewart, B. Malin, and J. Sun, "Paramo: A parallel predictive modeling platform for healthcare analytic research using elec-

- tronic health records,” *Journal of biomedical informatics*, vol. 48, pp. 160–170, 2014.
- [70] S. B. L. Barnett and T. A. Nurmagambetov, “Costs of asthma in the united states: 2002-2007,” *Journal of allergy and clinical immunology*, vol. 127, no. 1, pp. 145–152, 2011.
 - [71] O. J. Akinbami, J. E. Moorman, X. Liu, *et al.*, *Asthma prevalence, health care use, and mortality: United states, 2005-2009*. US Department of Health, Human Services, Centers for Disease Control, and Prevention, National Center for Health Statistics Washington, DC, 2011.
 - [72] J. S. Hughes, R. F. Averill, J. Eisenhandler, N. I. Goldfield, J. Muldoon, J. M. Neff, and J. C. Gay, “Clinical risk groups (crgs): A classification system for risk-adjusted capitation-based payment and health care management,” *Medical care*, vol. 42, no. 1, pp. 81–90, 2004.
 - [73] I. Guyon and A. Elisseeff, “An introduction to variable and feature selection,” *Journal of machine learning research*, vol. 3, no. Mar, pp. 1157–1182, 2003.
 - [74] H. Schütze, C. D. Manning, and P. Raghavan, *Introduction to information retrieval*. Cambridge University Press, 2008, vol. 39.
 - [75] R. G. Lomax and D. L. Hahs-Vaughn, *Statistical concepts: A second course*. Routledge, 2013.
 - [76] Y. Benjamini and Y. Hochberg, “Controlling the false discovery rate: A practical and powerful approach to multiple testing,” *Journal of the royal statistical society. Series B (Methodological)*, pp. 289–300, 1995.
 - [77] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, “Spark: Cluster computing with working sets,” *HotCloud*, vol. 10, no. 10-10, p. 95, 2010.
 - [78] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, “Liblinear: A library for large linear classification,” *Journal of machine learning research*, vol. 9, no. Aug, pp. 1871–1874, 2008.
 - [79] C.-C. Chang and C.-J. Lin, “Libsvm: A library for support vector machines,” *ACM transactions on intelligent systems and technology (TIST)*, vol. 2, no. 3, p. 27, 2011.
 - [80] L. Breiman, “Random forests,” *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.
 - [81] S. M. Pollart, M. D. Chapman, G. P. Fiocco, G. Rose, and T. A. Platts-Mills, “Epidemiology of acute asthma: Ige antibodies to common inhalant allergens as a risk

factor for emergency room visits,” *Journal of Allergy and Clinical Immunology*, vol. 83, no. 5, pp. 875–882, 1989.

- [82] H. Löhr, A.-R. Sadeghi, and M. Winandy, “Securing the e-health cloud,” in *Proceedings of the 1st ACM International Health Informatics Symposium*, ACM, 2010, pp. 220–229.
- [83] M. Li, S. Yu, K. Ren, and W. Lou, “Securing personal health records in cloud computing: Patient-centric and fine-grained data access control in multi-owner settings,” in *International conference on security and privacy in communication systems*, Springer, 2010, pp. 89–106.
- [84] M. Li, S. Yu, Y. Zheng, K. Ren, and W. Lou, “Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption,” *IEEE transactions on parallel and distributed systems*, vol. 24, no. 1, pp. 131–143, 2013.
- [85] G. Hripcsak, J. D. Duke, N. H. Shah, C. G. Reich, V. Huser, M. J. Schuemie, M. A. Suchard, R. W. Park, I. C. K. Wong, P. R. Rijnbeek, *et al.*, “Observational health data sciences and informatics (ohdsi): Opportunities for observational researchers,” *Studies in health technology and informatics*, vol. 216, p. 574, 2015.
- [86] G. Hripcsak, P. B. Ryan, J. D. Duke, N. H. Shah, R. W. Park, V. Huser, M. A. Suchard, M. J. Schuemie, F. J. DeFalco, A. Perotte, *et al.*, “Characterizing treatment pathways at scale using the ohdsi network,” *Proceedings of the National Academy of Sciences*, vol. 113, no. 27, pp. 7329–7336, 2016.
- [87] W.-Q. Wei and J. C. Denny, “Extracting research-quality phenotypes from electronic health records to support precision medicine,” *Genome medicine*, vol. 7, no. 1, p. 41, 2015.
- [88] M. D. Ritchie, J. C. Denny, D. C. Crawford, A. H. Ramirez, J. B. Weiner, J. M. Pulley, M. A. Basford, K. Brown-Gentry, J. R. Balser, D. R. Masys, *et al.*, “Robust replication of genotype-phenotype associations across multiple diseases in an electronic medical record,” *The American Journal of Human Genetics*, vol. 86, no. 4, pp. 560–572, 2010.
- [89] D. M. Roden and J. C. Denny, “Integrating electronic health record genotype and phenotype datasets to transform patient care,” *Clinical Pharmacology & Therapeutics*, vol. 99, no. 3, pp. 298–305, 2016.
- [90] R. J. Carroll, A. E. Eyler, and J. C. Denny, “Naïve electronic health record phenotype identification for rheumatoid arthritis,” in *AMIA annual symposium proceedings*, American Medical Informatics Association, vol. 2011, 2011, p. 189.

- [91] M. F. Davis, S. Sriram, W. S. Bush, J. C. Denny, and J. L. Haines, “Automated extraction of clinical traits of multiple sclerosis in electronic medical records,” *Journal of the American Medical Informatics Association*, vol. 20, no. e2, e334–e340, 2013.
- [92] G. Jiang, R. C. Kiefer, L. V. Rasmussen, H. R. Solbrig, H. Mo, J. A. Pacheco, J. Xu, E. Montague, W. K. Thompson, J. C. Denny, *et al.*, “Developing a data element repository to support ehr-driven phenotype algorithm authoring and execution,” *Journal of biomedical informatics*, vol. 62, pp. 232–242, 2016.
- [93] C. W. Yancy, M. Jessup, B. Bozkurt, J. Butler, D. E. Casey, M. H. Drazner, G. C. Fonarow, S. A. Geraci, T. Horwich, J. L. Januzzi, *et al.*, “2013 accf/aha guideline for the management of heart failure: A report of the american college of cardiology foundation/american heart association task force on practice guidelines,” *Journal of the American College of Cardiology*, vol. 62, no. 16, e147–e239, 2013.
- [94] V. L. Roger, S. A. Weston, M. M. Redfield, J. P. Hellermann-Homan, J. Killian, B. P. Yawn, and S. J. Jacobsen, “Trends in heart failure incidence and survival in a community-based population,” *Jama*, vol. 292, no. 3, pp. 344–350, 2004.
- [95] P. J. Guo and M. I. Seltzer, “Burrito: Wrapping your lab notebook in computational infrastructure,” 2012.
- [96] M Ragan-Kelley, F Perez, B Granger, T Kluyver, P Ivanov, J Frederic, and M Bussonnier, “The jupyter/ipython architecture: A unified view of computational research, from interactive exploration to communication and publication,” in *AGU Fall Meeting Abstracts*, 2014.
- [97] H. Shen, “Interactive notebooks: Sharing the code,” *Nature News*, vol. 515, no. 7525, p. 151, 2014.
- [98] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, “Gene selection for cancer classification using support vector machines,” *Machine learning*, vol. 46, no. 1-3, pp. 389–422, 2002.
- [99] R. J. Carroll, W. K. Thompson, A. E. Eyler, A. M. Mandelin, T. Cai, R. M. Zink, J. A. Pacheco, C. S. Boomersshine, T. A. Lasko, H. Xu, *et al.*, “Portability of an algorithm to identify rheumatoid arthritis in electronic health records,” *Journal of the American Medical Informatics Association*, vol. 19, no. e1, e162–e169, 2012.
- [100] P. B. Jensen, L. J. Jensen, and S. Brunak, “Mining electronic health records: Towards better research applications and clinical care,” *Nature Reviews Genetics*, vol. 13, no. 6, p. 395, 2012.

- [101] B. A. Chan and B. G. Hughes, “Targeted therapy for non-small cell lung cancer: Current standards and the promise of the future,” *Translational lung cancer research*, vol. 4, no. 1, p. 36, 2015.
- [102] Axovant. (2017). Axovant announces negative topline results of intepirdine phase 3 mindset trial in alzheimer’s disease, (visited on 05/07/2018).
- [103] F. Times. (2017). Axovant alzheimers drug fails in clinical trials, (visited on 05/07/2018).
- [104] C. for Medicare and M. Services. (2017). Medicare provider utilization and payment data, (visited on 05/07/2018).