Patient Clustering using Electronic Medical Records

by

Andrew Shea

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2020

Author .....
Department of Electrical Engineering and Computer Science
May 20, 2020

Certified by .....
Manolis Kellis
Professor, Computer Science
Thesis Supervisor

Accepted by .....
Katrina LaCurts
Chair, Master of Engineering Thesis Committee

Patient Clustering using Electronic Medical Records

by

Andrew Shea

Submitted to the Department of Electrical Engineering and Computer Science
on May 20, 2020, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

Electronic health records (EHR) and their wealth of patient health information present new opportunities for understanding relationships between patients and their conditions. However, EHR data sparsity, quality, and accessibility present various computational challenges. To address these challenges, we apply spectral clustering and variational autoencoders to obtain compact patient representations and clusters from EHR in an unsupervised manner. We apply these methods to the MIMIC dataset, from which we only use ICD-9 diagnostic codes to ensure data accessibility. After obtaining clusters, we conduct high-resolution analysis by examining the 5 most frequent phenotypes within each cluster. We then conduct low-resolution analysis by examining the distribution of phenotypes within each cluster, examining the relationships amongst the most prevalent phenotypes in each cluster by constructing a cluster network, and comparing our findings to existing medical literature. While preliminary, these results suggest that learning from sparse EHR data is sufficient for uncovering associations between conditions and diseases.

Thesis Supervisor: Manolis Kellis
Title: Professor, Computer Science

Acknowledgments

I would like to start by thanking Dr. Manolis Kellis for the opportunity of being a part of the Computational Biology Group and for the guidance and support throughout the completion of this thesis. I've learned a great deal about science and leadership from our regular meetings and will be taking these lessons with me to the future.

I would also like to thank Yongjin Park and the entire Kellis Lab for the mentorship and good times. To Yongjin, thank you for advising me throughout this project and for all the life lessons. To the entire Kellis Lab, thank you for the exceptionally welcoming, supportive, and fun environment.

Thank you to the mentors who have helped me make it this far. To individuals, special thanks to Christine Pilcavage, Dianbo Liu, Jasper Degens, Duc Pham, Tian He, and Costa Christopoulos. To organizations, special thanks to MISTI Japan, Imobilare, Knockout Kings, and teamLab. And to everyone else, you know who you are. Here's to the future.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The analysis of electronic health records (EHR) — an electronic version of a patient's medical history — has led to innovations in healthcare and research, especially in machine learning whose methods are well-suited for digesting large quantities of data. Spurred by the Health Information Technology for Economic and Clinical Health Act of 2009 which provided incentives for hospitals to implement healthcare information technology systems, EHR have seen widespread adoption in the United States where a vast majority of hospitals have some form of an EHR system [3]. EHR themselves may include a variety of clinical data such as a patient's demographics, progress notes, medications, diagnostics, vital signs, and more [1]. Much of these data are standardized into systematic codes, such as the International Classification of Diseases (ICD)-9 which is used to track patient diagnoses and procedures. These increasingly prevalent EHR enable researchers not only to further research in machine learning and medicine, but also gain greater insights into diseases and their relationships. In this thesis, we utilize EHR to tackle two areas of research, namely patient representation learning and clustering.

The vast amount of EHR data on patient diagnostics enable researchers to examine relationships between patients, conditions, and diseases on in a broader, more holistic manner. This could potentially lead researchers to discover novel associations between diseases and conditions that previously had been unexplored or overlooked. Such an ability would have tangible impacts in areas such as comorbidity analysis and

symptom clustering which currently focus on finding disease associations with respect to a single disease rather than more broadly (see Chapter 2, Related Works).

In spite of these opportunities and advances, analyzing EHR remain challenging in machine learning due to their quality and availability. EHR can often be high-dimensional, sparse, incomplete, noisy, and error prone [30, 69]. These qualities make it difficult for machine learning methods to learn patterns that generalize well across patients and datasets [6]. Additionally, EHR data can often be inaccessible, such as in cases where researchers access private data through partnerships with specific hospitals or companies. While some researchers have tackled this issue of accessibility by making comprehensive datasets publicly available [48, 31, 57], the overall lack of publicly available data still poses a challenge in ensuring that methods are reproducible, especially when certain data types may be present in one dataset but absent in another.

One promising avenue of machine learning research to address these inherent challenges is to automatically obtain patient representations or compact representations of patient data that still retain key characteristics of the original data. While early research on patient representation relied on domain experts and time-consuming manual feature selection [57], more recent approaches utilize machine learning methods which are able to automatically identify patterns in data to produce compact yet meaningful representations of the original data (see Chapter 2, Related Works).

To address these challenges of data quality and availability inherent to EHR while also examining diseases on a broader scale, we apply a modified version of spectral clustering and variational autoencoders (VAE) to obtain patient representations. We then cluster these patient representations to gain insights into the patients' underlying conditions. In our modified version of spectral clustering, we compute our affinity matrix using only a subset of mutual nearest neighbors to simulate a scenario where data is limited, select eigenvector features and patient representations based on empiricism rather than theoretical heuristics, and make slight computational adjustments to account for computational constraints in our eigensolvers. We apply this method to one of the largest, freely available datasets, Medical Information Mart for Intensive Care

(MIMIC)-III. From this dataset, we only use International Classification of Disease (ICD)-9 codes as data, ensuring that our data types are accessible in most public and private datasets. We assess our patient representations and clustering first by examining the most frequently occurring ICD-9 codes within each cluster and then by conducting a higher-level analysis using a hybrid Phecode-ICD feature representation.

Chapter 2

Related Works

2.1 Patient Representation Learning

Extensive research exists on improving healthcare applications using EHRs and machine learning (for surveys see [68, 51, 45, 12]). Among these, researchers have also applied machine learning techniques to learn high-level patient representations from high-dimensional and often noisy patient data. Miotto et al. learned a general representation that outperformed previous representations on disease classification and patient disease tagging using a three-layer stack of denoising autoencoders and aggregated EHR data from 700,000 patients from the Mount Sinai data warehouse [44]. Suo et al. learned patient representations and used them to measure pairwise patient similarities using a convolutional neural network and data from 9,528 patients from a larger, real world dataset [58]. Other researchers have leveraged time-series patient data to learn representations. Ma et al. learned patient representations and demonstrated their efficacy in predicting the onset of coronary heart failure using temporal patient data extracted from proprietary and public datasets and a model consisting of attentive and time-aware modulars and a hybrid network comprising recurrent and convolutional neural networks [42]. Lyu et al. demonstrated the efficacy of their unsupervised representations through practical applications using the eICU Collaborative Research Database [48] and sequence-to-sequence models [41]. As the research on representation learning accelerates, more recent works have addressed is-

sues relating to data-silos and lack of interoperability between healthcare centers by learning patient representations in a distributed manner [40, 70, 29, 37].

2.2 Symptom Clustering and Comorbidity Analysis

While considerable literature exists on comorbidity analysis and symptom clustering, where researchers use data-driven approaches to identify co-occurring symptoms and patient subgroups, most works start by selecting for a specific phenotype rather than using a patient population with a broad range of conditions. For example, in early research conducted in symptom clustering, Sanders et al. used multidimensional cluster analysis to identify patient subgroups in 180 patients suffering from chronic pain [53], Knishkowy et al. analyzed survey data to identify symptom clusters from a subset of 259 Israeli school children with recurrent psychosomatic symptoms [36], and Ho et al. used principal component analysis followed by varimax rotation to identify 5 symptom clusters from a randomly surveyed group of perimenopausal women [27]. More recent literature on symptom clustering focuses on various stages and forms of cancer [15, 34, 19, 5]. For example, Walsh et al. used an agglomerative hierarchical clustering approach to examine 25 symptoms on 922 patients with advanced cancer and identified 7 symptom clusters which were fatigue: anorexia-cachexia, neuropsychological, upper gastrointestinal, nausea and vomiting, aerodigestive, debility, and pain [65]. Tsai et al. used exploratory factor analysis on survey data obtained from 427 patients with advanced cancer to identify 5 symptom clusters which were loss of energy, poor intake, autonomic dysfunction, aerodigestive impairment, and pain complex [62]. This also holds true for comorbidity analysis, which has examined symptoms in connection with hypertension [10, 63, 33] among others.

2.3 Large-scale Comorbidity Analysis

To the best of our knowledge, only a handful of studies have attempted to draw associations between a wide array of conditions using machine learning and clinical data.

Li et al. developed a Bayesian unsupervised learning approach based on collaborative filtering and latent topic models to identify 100 groups of diseases from over 50,000 EHR features [38]. Guo et al. analyzed data from over 8,000,000 patients in 453 hospitals in China to build a disease comorbidity network with 5,702 nodes and 258,535 edges [23]. It's worth noting several works, such as that by Goh et al. [20], have built networks across thousands of disease genes, however these works typically do not utilize both machine learning and clinical data.

Chapter 3

Spectral Clustering using Approximate Mutual Nearest Neighbors

In spectral methods for clustering, we build a matrix by computing distances between points and then use the top eigenvectors of this matrix to cluster data points. Various spectral methods for clustering exist and have been used across various domains. Here, we present a spectral clustering method based on [46].

Representing data points as distances or similarities to other data points is beneficial as doing so densifies sparse datasets and allows data representations to generalize across data types and datasets. For example, if we are given a dataset that is high dimensional and sparse then representing each data point as a set of distances to other data points can drastically increase the density of the dataset. Likewise, by representing data points as a set of distances rather than their original features, we abstract away individual data types to create a representation that is potentially more general. This representation of spectral methods is then especially relevant to healthcare, where data on patients is often sparse and can vary greatly between hospitals and insurance systems.

3.1 Methods and Materials

3.1.1 Dataset: Medical Information Mart for Intensive Care (MIMIC)-III

The Medical Information Mart for Intensive Care (MIMIC)-III is a freely accessible, large database containing information relating to patients admitted to intensive care units and includes data such as vital signs, medications, observations and notes, diagnostic codes, and more [31]. The data was collected from the Beth Israel Deaconess Medical Center in Boston, Massachusetts between 2001 and 2012. Because patients were admitted to intensive care units, MIMIC-III contains more data pertaining towards patients who are critically ill or injured rather than, for example, routine check up data [31].

International Classification of Disease (ICD)-9 codes

While MIMIC-III contains a wide variety of patient data, the patient data we utilize here are diagnostic codes which are encoded using standardized International Classification of Disease (ICD)-9 codes [31]. For example, the ICD-9 code "001" corresponds to Cholera. In total, MIMIC-III contains data for 46,520 distinct patients including adults (defined to be aged 16 years or above) and neonates (ie. newborn children). Each patient has ICD-9 codes corresponding to each of their visits to the intensive care unit. In aggregate, we find there are 6,984 ICD-9 codes within MIMIC-III.

While many data types exist in MIMIC-III we use ICD-9 diagnostic codes due their widespread usage in hospitals in the US and abroad. ICD-9 codes are an internationally recognized classification system which are primarily used for hospital administrivia [60]. While the ICD-9 has since been revised and many countries are now using or in the process of converting to ICD-10 [8, 25], ICD-9 still encodes core features and diagnostic information and can be converted to ICD-10 [18].

ICD diagnostic codes are useful for clinical analysis and tracking disease despite some of their potential inaccuracies. For example, some studies have noted that ICD-

9's are inaccurate for tracking some diseases, such as one study that showed ICD-9 codes were inaccurately assigned to patients experiencing traumatic brain injury [4]. Nonetheless, other studies have shown the use of ICD-9 codes to be accurate in tracking certain diseases [54, 21, 7, 17].

3.1.2 Modified Spectral Clustering

Given a set of points $S = \{s_1, ...s_n\}$ in $\mathbb{R}^l$ that we want to cluster into k subsets:

1. Build an Approximate Nearest Neighbors tree T using the random-projection based method of Locality Sensitive Hashing (LSH) to approximate the cosine distance between all pairs of points, where the cosine distance between points s_i and s_j is

   $$d(s_i, s_j) = \frac{s_i \cdot s_j}{||s_i||||s_j||}.$$

   We use a publicly available library called Annoy (Approximate Nearest Neighbors Oh Yeah) from Spotify.

2. Using the precomputed distances from T, build the affinity matrix $A' \in \mathbb{R}^{n \times n}$. That is,

   $$A'_{ij} = \exp(-C * d(s_i, s_j)^2)$$

   $$C = 1/2\sigma^2$$

   if and only if s_j is an m nearest neighbor of s_i and 0 otherwise. in our experiments, we found that $C = 2$ was sufficient. Additionally, note that A' is not necessarily symmetric and each row of A' should contain exactly m nonzero values.

3. Compute the symmetric matrix A from A' by keeping nonzero values of A' if and only if s_i and s_j are mutual nearest neighbors. That is, $A_{ij} = A'_{ij}$ iff s_i and s_j are mutual nearest neighbors. Otherwise, $A_{ij} = 0$. Note that A is symmetric and nonuniform.

4. Using A, compute the diagonal matrix D where D_{ii} is the sum of A's ith row.

5. Compute $L = D^{-1/2}AD^{-1/2}$.

6. Form the matrix X by finding the p largest eigenvectors $x_1, x_2, ..., x_p$ of L, where $X = [x_1, x_2, ..., x_p] \in \mathbb{R}^{n \times p}$.

7. Compute the matrix Y by renormalizing the rows of X to have unit length. That is,
$$Y_{ij} = \frac{X_{ij}}{(\sum_j X_{ij}^2)^{-1/2}}$$

8. Treating each row of Y as a point in $\mathbb{R}^p$, use K-means to cluster the data into k clusters.

9. Assign the original point s_i to cluster j if only if row i for Y was assigned to cluster j.

Changes to the original spectral clustering algorithm

Our method differs from standard spectral clustering in several ways:

- Our method leverages an Approximate Nearest Neighbors data structure. While we sacrifice some precision in the process, this modification allows our method to scale better with larger datasets.

- We use cosine distance in the RBF Kernel instead of euclidean distance. Using cosine distance not only performs well empirically and simplifies implementation, but also has been shown to perform similarly to euclidean distance in high dimensional space [50].

- We compute a partial affinity matrix A using mutual nearest neighbors whereas the original version in [46] forms A using all pairwise distances between points. We choose to compute a partial A in order to simulate a real-world healthcare scenario where data may be missing or lacking or where extensive comparisons may not be a viable option.

- We treat each row in Y as a point in $\mathbb{R}^p$ and use K-means to obtain k clusters where k does not necessarily equal p. In the original method, the number of clusters k equals the number of selected eigenvector features p. We choose to use different values given our empirically successful results.

Intuition behind spectral clustering

Here we briefly provide intuition for why spectral methods for clustering perform well. Note that numerous resources explore the underlying mechanisms in greater detail using a variety of methods such as idealized cases, comparisons to other algorithms, mathematical proofs, and more [13, 28, 46, 55, 64].

Intuitively, spectral methods for clustering work because data points in Cartesian space can sometimes be partitioned better in similarity space. We show this by considering an idealized case. Assume each data point $s_1, ..., s_n$ belongs to one of p distinct clusters and each cluster is separated (ie. infinitely far) from all other clusters. We start with our original dataset S, where each data point exists in Cartesian space, and compute $L \in \mathbb{R}^{n \times n}$, where each data point can be thought of as existing in similarity space. Next, we compute L's eigenvector matrix $X \in \mathbb{R}^{n \times p}$, where $p < n$ without loss of generality.

In this idealized case, L would be a block-diagonal matrix, leading to a convenient property where each block corresponds to a unique cluster [46]. This is because each of the p columns (ie. eigenvectors) in L's eigenvector matrix $X \in \mathbb{R}^{n \times p}$ is orthogonal to each other such that each row (ie. transformed data point) will contain a single nonzero value [46]. In other words, each row in X will lie along one of p unique axes corresponding to one of p distinct clusters which can then be identified through simple inspection or K-Means. And because each row in X corresponds to a data point in the original dataset S, each data point $s_1, ..., s_n$ can then be mapped accordingly to one of p distinct clusters.

While the idealized case 1) relies on L being a block-diagonal matrix and 2) makes assumptions about the dataset S that may not necessarily hold in real world datasets, spectral methods for clustering still perform well empirically as they hold

certain provable properties which ensure nontrivial partitioning of the original data [46, 55].

3.1.3 Cluster Analysis

We assess the quality of our patient clustering by 1) examining the most frequent ICD-9 codes within individual sets of clusters and 2) conducting a high-level analysis over all clusters. By examining individual clusters, we are able to make a higher-resolution assessment of how well our clusters capture meaningful conditions. By examining all clusters together, we are able to make a higher-level assessment of the general trends and conditions our clusters capture.

Examining most frequent symptoms in individual clusters

To assess individual sets of clusters, we separate clusters into 3 classes by size — small, medium, large — and then examine the 5 most frequently occurring conditions within each cluster. The conditions are encoded based on International Classification of Disease (ICD)-9 codes (see Section 3.1.1). Examining the diseases using the ICD-9 codes allows us to gain a high resolution understanding of the specific conditions within a cluster. To decide the small, medium, and large size thresholds, we obtain the interquartile range (IQR) of cluster sizes and define small clusters to be below the IQR, medium clusters to be within the IQR, and large clusters to be above the IQR. We then analyze only a subset of the cluster due to the potentially large number of clusters that result from K-Means. We also only examine the most frequent conditions within each cluster, as some cluster may contain several hundred different ICD-9 codes.

High-level analysis of all clusters

Hybrid Phecode-ICD representation. To assess the quality of all clusters, we start by 1) using a hybrid encoding consisting of Phecodes and ICD-9 codes rather than using ICD-9 codes alone to represent patient features (we refer to the hybrid encoding as

Phecode-ICD), 2) obtaining the distribution of Phecode-ICD features for each cluster, and 3) using the new Phecode-ICD representation to construct a network across the clusters. Similar to the ICD-9 codes, Phecodes are a hierarchical coding structure used for encoding phenotypes that can be converted to ICD-9 [67, 14]. For example, the ICD-9 code "001" which corresponds to "Cholera" maps to the Phecode "008" corresponding to "Intenstinal infection" [14]. A major utility for using Phecodes over ICD-9 for high-level analysis is that Phecodes can be mapped to higher-level disease categories that still preserve a high-degree of detail [67]. For example, ICD-9 codes "001.0" and "002" map to "Cholera due to Vibrio cholerae" and "Typhoid and paratyphoid fevers," respectively but both codes map to Phecode "008" corresponding to "Intenstinal infection" [14].

We perform our mapping by mapping specific ICD-9 codes to the nearest integer Phecode, as doing so enables us to obtain a disease class that is broad yet still retains a high level of detail. For example, the ICD-9 code "002.0" originally maps to the Phecode "008.5." Due to the hierarchical structure of Phecodes, we would then map the ICD-9 code to "008."

In instances where we were unable to map between ICD-9 to Phecode using our method and online resources [14], we retained the original ICD-9. For example, we were unable to find a direct Phecode mapping for ICD-9 code "V3000" corresponding to "Single liveborn; born in hospital; delivered by cesarean section." After converting our from ICD-9 codes hybrid Phecode-ICD, we then obtain a feature distribution for each cluster.

Weighted Jaccard similarity network. After obtaining Phecode-ICD cluster-level distributions, we construct an inter-cluster similarity network using weighted Jaccard similarity. We compute this metric for all pairs of clusters and then build a similarity graph. The weighted Jaccard similarity value is a metric which measures the similarity between two discrete sets In our case, these sets are our clusters and each element in a set is a Phecode-ICD feature. Mathematically, let $J(A, B)$ be the weighted Jaccard similarity value between clusters A and B. Additionally, let i denote some Phecode-

ICD feature and A_i denote its frequency in cluster A and B_i denote its frequency in cluster B. The weighted Jaccard similarity value is then defined as

$$J(A, B) = \frac{\Sigma_i min(A_i, B_i)}{\Sigma_i max(A_i, B_i)}.$$

Silhouette plots for qualitative cluster assessment. We use silhouette analysis to qualitatively assess whether our selected number of clusters is appropriate. That is, we examine the average silhouette score across all clusters while inspecting the corresponding silhouette plot. We do so using the eigenvector patient representations.

A silhouette score is a value between -1 and 1 which measures of how similar an object is to its own cluster compared to other clusters [66]. The average silhouette score across all clusters is then calculated by averaging the score across all data points. In our setting, we compute a patient's silhouette score by computing the average distance between that patient and all other patients within the same cluster. Next, we compute the average distance between that patient and all other patients within the *nearest* cluster. Finally, we then compute a ratio using the two values.

Mathematically, let patient i belong to cluster C_i and its silhouette score be $s(i)$. Next let $d(i, j)$ be the euclidean distance between patient i's eigenvector representation and some patient j's eigenvector representation, $a(i)$ be the average *intra*-cluster distance for patient i, and $b(i)$ be the average *inter*-cluster distance between patient i and all points in the cluster nearest to cluster C_i. The silhouette score is then

$$a(i) = \frac{1}{|C_i| - 1} \sum_{j \in C_i, i \neq j} d(i, j)$$

$$b(i) = \min_{k \neq i} \frac{1}{|C_k|} \sum_{j \in C_k} d(i, j)$$

$$s(i) = \begin{cases} \dfrac{b(i) - a(i)}{\max\left(a(i), b(i)\right)}, & \text{if } |C_i| > 1 \\ 0, & \text{if } |C_i| = 1 \end{cases}$$

Based on the formula for $s(i)$ above, it can be shown that a silhouette score is within the range -1 and 1. Intuitively, lower silhouette scores represent ambiguous or potentially inappropriate cluster assignments while higher values represent decisive or distinct cluster assignments. Considering a toy example, if a data point is located within a distinct cluster that is well-separated from neighboring clusters, then we expect the data point's *inter*-cluster distance $b(i)$ to be greater than its *intra*-cluster distance $a(i)$, resulting in a positive silhouette score. On the other hand, if a data point exists at the boundary of two potential clusters that have a similar density, then we expect this data point's *inter*-cluster distance $b(i)$ and *intra*-cluster distance $a(i)$ to be approximately equal, resulting in a silhouette score that is closer to 0. And in certain cases, such as when a data point is assigned to a sparse cluster with low density while a dense cluster is nearby, then its *inter*-cluster distance $b(i)$ may be less than its *intra*-cluster distance $a(i)$, resulting in a negative silhouette score. Generalizing this understanding to an entire set of clusters, we would then expect a good clustering to have a positive average silhouette score.

3.2 Experiment

3.2.1 Preprocessing: Binary ICD Matrix

We compute a binary 0-1 matrix where each patient is represented by a binary vector and a 1 represents the presence an ICD-9 diagnostic code. We do so by first opening the "diagnoses_icd" table in MIMIC-III. Prior to preprocessing, this table has 651,047 rows with 5 columns. We only use 2 of the 5 columns that correspond to the patient identification number ("SUBJECT_ID") and the ICD-9 diagnostic code column ("ICD9_CODE"). We then aggregate this information into a binary matrix, such that each row or vector corresponds to a single patient and each column corresponds to a single ICD-9 diagnostic code. Doing so results in a matrix of shape 46,520 patients $\times$ 6,984 ICD-9 diagnostic codes. Note that while there are over 14,000 ICD-9 codes, we found that only 6,984 are present in the "diagnoses_icd" table of

MIMIC-III.

3.2.2 Modified Spectral Clustering

Setting up the Approximate Nearest Neighbor Tree T

After building the binary matrix, we then pass this into our spectral clustering algorithm (see Section 3.1.2) and start by building an approximate nearest neighbor tree T. We do so using the open-source library "Approximate Nearest Neighbor Oh Yeah" (ANNOY) from Spotify, which creates an approximate nearest neighbor tree using the random projection method of Locality-Sensitive Hashing [11]. In the approximate nearest neighbor tree data structure, we are able to obtain the nearest n neighbors for any given patient in sublinear time. This means that over the entire dataset, we are able to efficiently compute our affinity matrix in under $O(n^2)$ runtime. Using the ANNOY library, we construct the approximate nearest neighbors tree using default parameters, cosine distance, and 10 intermediate trees (a parameter where higher values increase the precision of nearest neighbor outputs at the expense of runtime for constructing the tree).

Computing initial affinity matrix A'

Next, we build a preliminary affinity matrix A'. To start, for each patient s_i, we query our approximate nearest neighbor tree T to obtain patient s_i's $n = 50$ nearest neighbors. While we tested other values for the number of nearest neighbors n, we highlight $n = 50$ as this was the smallest value that yielded efficacious results. Then, for each nearest neighbor, we compute the affinity metric according to the modified Radial Basis Function kernel in Section 3.1.2, step 2. After repeating this process for each patient, we obtain A', a 46,520 patient $\times$ 46,520 patient affinity matrix where each row has exactly $n = 50$ nonzero values.

Computing A from A' by only keeping mutual nearest neighbors

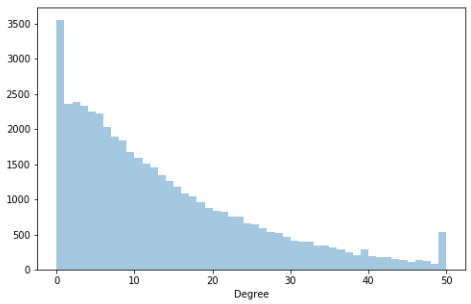
After obtaining A', we build A by only keeping values corresponding to mutual nearest neighbors. We perform this step because the nearest neighbors obtained from the approximate nearest neighbor tree T are not necessarily symmetric and obtaining a symmetric affinity matrix A is necessary for this spectral clustering method. Two patients are considered mutual nearest neighbors if both patients are in each other's neighborhoods. More specifically, for any pair of patients s_i and s_j, their corresponding affinity values A_{ij} and A_{ji} are nonzero if A'_{ij} and A'_{ji} are also nonzero and 0 otherwise. As a result, the patients in A have a non-uniform distribution as shown in Figure 3-1a. From this distribution, note that there are two "spikes" in the lowest and highest bins. These peaks are caused by an excess of patients with either 0 or $n = 50$ neighbors. Based on experiments, we found the presence of these patients biased the final output and so we removed them. Doing so results in a final A matrix of shape 42,517 patients $\times$ 42,517 patients, where each row was between 1 and 49 (inclusive) nonzero values. The updated distribution is shown in Figure 3-1b, where the peaks in the lowest and highest bins have now been reduced or removed.

Computing diagonal matrix D and normalized Laplacian L

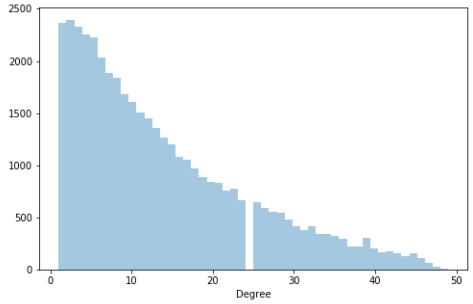
With our A matrix, we then straightforwardly compute D and L according to Section 3.1.2 steps 4 and 5. We note that care should be taken when computing D and L, as some Python libraries can result in *nan* or "not a number" values, such as when taking the libraries mistakenly take the inverse of 0.

Obtaining eigenvector feature matrix X

After obtaining L, we compute the top k eigenvector matrix X and then straightforwardly normalize its rows to unit norm to obtain Y. In terms of implementation, there are several ways of obtaining X from L. For example, some methods involve computing the eigenvector matrix in its entirety and then selecting the top k eigenvectors based on the quality of results; this method allows for a potentially more



(a) Distribution of node degrees in affinity matrix A *before* removing nodes with 0 and 50 connections.



(b) Distribution of node degrees in affinity matrix A *after* removing nodes with 0 and 50 connections. Note that by removing patients with 50 connections, we also reduced the degrees of their neighbors. The empty bin in the middle contained patients who were tightly connected with patients with 50 connections.

Figure 3-1: Degree distributions of A before and after removing nodes. Note these distributions differ from that in standard spectral clustering [46], where the graph would be fully connected and each node would therefore have the same degree.

comprehensive evaluation of L's eigenvectors at the (potentially large) expense of computation time. Other methods allow us to compute the top k eigenvectors directly via truncated singular value decomposition (SVD), sacrificing some precision in exchange for reduced runtime. In our case, computing all eigenvectors of L using standard eigenvector solvers proved to be prohibitively expensive. As a result, we compute the top k eigenvectors directly. To ensure we examine a sufficient number of eigenvectors, we compute the top $k = 2000$ eigenvectors.

Computing X using truncated SVD. More specifically, we obtain the eigenvector matrix X from L by performing a truncated singular value decomposition (SVD) on L using the Python library scikit-learn function "randomized_svd," a function that runs accurately and efficiently [24, 39, 59]. Truncated singular value decomposition is often used for approximating matrices and, in our setting, is a convenient way of

computing X directly from L where our result is

$$L \approx X\Lambda X^T$$

where Λ is a sorted diagonal matrix containing L's top k eigenvalues and X's columns contain the corresponding eigenvectors.

Background on SVD and truncated SVD. To better understand truncated singular value decomposition, we first need to look at standard singular value decomposition. In standard singular value decomposition, we decompose some $n \times n$ real matrix M into

$$M = U\Sigma V^T$$

where U is a $n \times n$ real matrix such that $U^TU = I$, Σ is a $n \times n$ diagonal matrix with non-negative real numbers (ie. M's "singular values") on the diagonal, V is a $n \times n$ real matrix such that $V^TV = I$, and U and V^T are orthonormal matrices.

If M is a $n \times n$ symmetric real matrix, then singular value decomposition is equivalent to spectral- or eigen-decomposition. That is, through standard singular value decomposition, we can decompose our symmetric real matrix M into

$$M = U\Sigma V^T = E\Lambda E^T$$

where E is an $n \times n$ matrix whose columns are orthonormal eigenvectors of M and Λ is a $n \times n$ diagonal matrix containing M's eigenvalues.

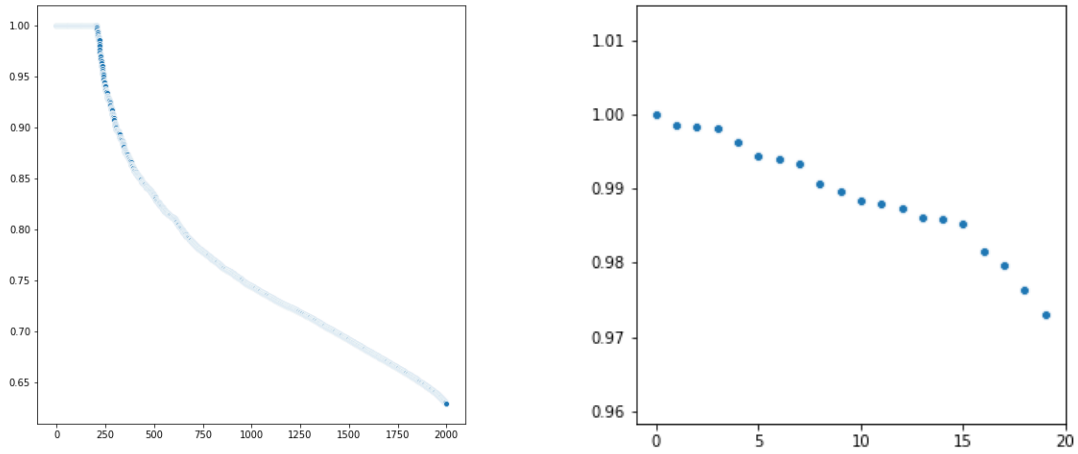
Truncated singular value decomposition then simply computes the top k eigenvectors and eigenvalues of M, meaning we would only keep the first k columns of the eigenvector matrix E and corresponding top k values in the eigenvalue matrix Λ. Doing so results in an approximate decomposition where

$$M \approx E_{1:k}\Lambda_{1:k}E_{1:k}^T$$

where $E_{1:k}$ is a $n \times k$ matrix and $\Lambda_{1:k}$ is a $k \times k$ matrix.

Quality control: selecting a subset of eigenvectors from X

Before computing the normalized Y from X, we first perform quality control by examining the eigenvalues of L, as the eigenvalues theoretically reveal much of the underlying structure of our data and influence which of L's eigenvectors we should select to build X. For example, if we examine L's first 2,000 eigenvalues in Figure 3-2a, we see that all eigenvalues are less than or equal to 1 and the eigenvalue 1 is repeated (we verify that the eigenvalue 1 is repeated approximately 200 times). This is worth noting, as the number of times the eigenvalue 1 is repeated should be approximately equal to the number of underlying clusters in our data and that, theoretically, we should be using these 200 eigenvectors to build X [46].



(a) First 2000 eigenvalues of L. The eigenvalue 1 is repeated ~ 200 times.

(b) Eigenvalues 200 - 220 of L corresponding to the eigenvectors we use to build X.

Figure 3-2: Sorted eigenvalues of L matrix

However, we arrive at a different approach and instead compose X using L's first 20 eigenvectors starting from 203, where 203 is selected as it is approximately where the eigenvalues stop repeating. These eigenvalues can be seen in Figure 3-2b. We should note that several methods have been proposed for selecting the eigenvectors with varying degrees of efficacy, trade-offs and theoretical justifications [47, 32, 56].

Normalizing rows of X to obtain Y

After constructing our $45,217 \times 20$ matrix X using 20 eigenvectors of L, we obtain Y by normalizing X's rows.

K-Means Clustering using Y

After construbting Y, we then performing K-Means clustering where the number of clusters $K = 30$. Again, we note $K = 30$ does not equal the theoretical number of clusters 200 [46]. However, we see empirically that doing so yields better results.

Summary

To summarize our methods so far, we first built a cosine distance approximate nearest neighbor tree using the "Approximate Nearest Neighbors Oh Yeah" library from Spotify. Next, we construct our affinity matrix A' where for each patient, we compute the affinities using 50 approximate nearest neighbors and leave all other values 0. Then, we obtain A by examining each pair of patients in A' with a nonzero affinity value and keeping that value if and only if that pair are mutual nearest neighbors. We then use A to compute D and L and obtain L's eigen-decomposition using truncated singular value decomposition. We then select 20 of L's eigenvectors to create X, normalize the rows of X to obtain Y, and then use Y to run K-Means clustering with 30 clusters.

3.2.3 Cluster Analysis

Examining most frequent symptoms in individual clusters

We examine the quality of specific clusters by computing the interquartile range (IQR) of cluster sizes. From our experiment, we obtain an IQR of 74.25-2448.0 patients, which we round to 74-2448 patients. The IQR divides the clusters into 3 classes — small, medium, large — where small clusters are below the IQR, medium clusters are within the IQR, and large clusters are above the IQR. To analyze the ICD-9 codes, for each cluster we obtain the top 5 most frequently occurring ICD-9 codes. We then

normalize the probability distribution with respect to the top 5 ICD-9 codes and plot them in their respective heatmaps. The results can be seen in Figure **??**.

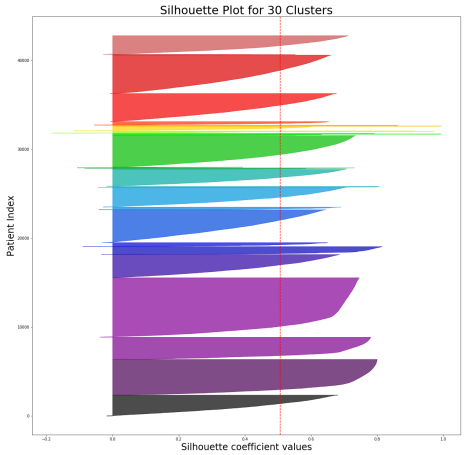
High-level analysis of all clusters

Hybrid Phecode-ICD representation. To gain a broader view of what our clusters capture, we first convert from ICD-9 codes to our hybrid Phecode-ICD representation. Doing so results in a reduction of features from 6,894 ICD9 codes to 1,110 Phecode-ICD features, where some ICD-9 codes were kept if they did not have a corresponding Phecode (see Section 3.1.3). We obtained the ICD9 to Phecode mapping using the map provided by the publicly available at phewascatalog.org [14]. We then obtain a sorted distribution of Phecodes and ICD9 codes for each cluster as seen in Figure 3-7a. Note the main significance of these plot is the shape of the sorted distribution and not the ordering.

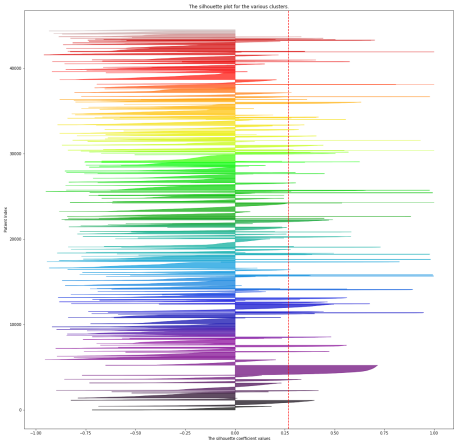
Weighted Jaccard similarity network. To create a network across our clusters, after reducing our original feature encoding from 6,894 ICD9 codes to 1,110 Phecodes and ICD9 codes, we obtain Jaccard similarity scores between all clusters. To obtain a single feature vector to represent each cluster, we sum over all patient Phecode-ICD features in that cluster. Then for each cluster, we compute pairwise similarities with all other clusters.

Silhouette plots for qualitative cluster assessment. To compute the silhouette scores, we start with our normalized eigenvector matrix Y, where each row contains a patient representation, as well as the cluster assignment results from our K-Means clustering, where $k = 30$. Then, for each patient, we compute the silhouette score and then plot it along side its cluster. This plot is shown in Figure 3-3a, where each "peak" corresponds to a cluster. Clusters are plotted according to their index assignment based on our K-Means clustering.

3.3 Results



(a) Empirically derived silhouette plot. This plot uses 20 eigenvectors corresponding to the 200-220 highest eigenvalues and K-Means with 30 clusters.



(b) Silhouette plot using standard spectral clustering heuristic. This plot uses 200 eigenvectors corresponding to the 200 first repeated eigenvalues and K-Means with 200 clusters.

Figure 3-3: Eigenvalue selection heuristic comparison: silhouette plots

Before examining the contents of our clusters, we first note that the silhouette plot corresponding to our method yields a qualitatively better silhouette plot than that corresponding to the standard spectral clustering heuristic. We see this in Figure 3-3 which shows a side-by-side comparison of silhouette plots produced using our empirical versus the standard heuristic for selecting eigenvectors. Generally, a silhouette plot for a set of clusters is considered well-formed if the vast majority of individual data points have positive silhouette scores and each cluster's silhouette plot is approximately smooth in shape with minimal spikes, as these spikes may indicate ambiguous or potentially improperly assigned data points [52]. Indeed, examining the silhouette plots in Figure 3-3, we see that our method obtains a well-formed silhouette plot with the exception of a few clusters who have much higher silhouette scores and a few patients with negative scores. Examining the plot produced using the standard heuristic, we see that a majority of patients actually have negative silhouette scores,

suggesting inappropriate clustering. While our heuristic yields qualitatively improved results, there may be other heuristics to consider (See Discussion).

3.3.1 Examining the most frequent symptoms in individual clusters

We also demonstrate that our clusters capture different subsets of symptoms, as shown in Figures 3-4, 3-5, and 3-6. We've divided the clusters based on size into small, medium, and large clusters and show that in some cases our clusters capture distinct sets of diagnostic ICD-9 codes and in other cases capture overlapping ICD-9 codes. For example, in Figure 3-4 we see that cluster numbers 1, 2, 3, and 18 capture distinct sets of ICD-9 codes (as evidenced by the diagonal). At the same time, clusters 4, 26, 27, and 29 capture similar groups of ICD9 codes corresponding to coronary atherosclerosis, hypertension, hyperlipidemia, atrial fibrillation, and congestive heart failure. In our medium-sized clusters in Figure 3-5, we observe a similar pattern where more overlap in ICD-9 codes commonly associated with childbirth such as cesarean section, prophylactic vaccination, and circumcision. Lastly, our small clusters shown in Figure 3-6 have more overlapping ICD-9 codes.

Figure 3-4: Top 5 ICD-9 codes for Large Clusters

Figure 3-5: Top 5 ICD-9 codes for Medium Clusters

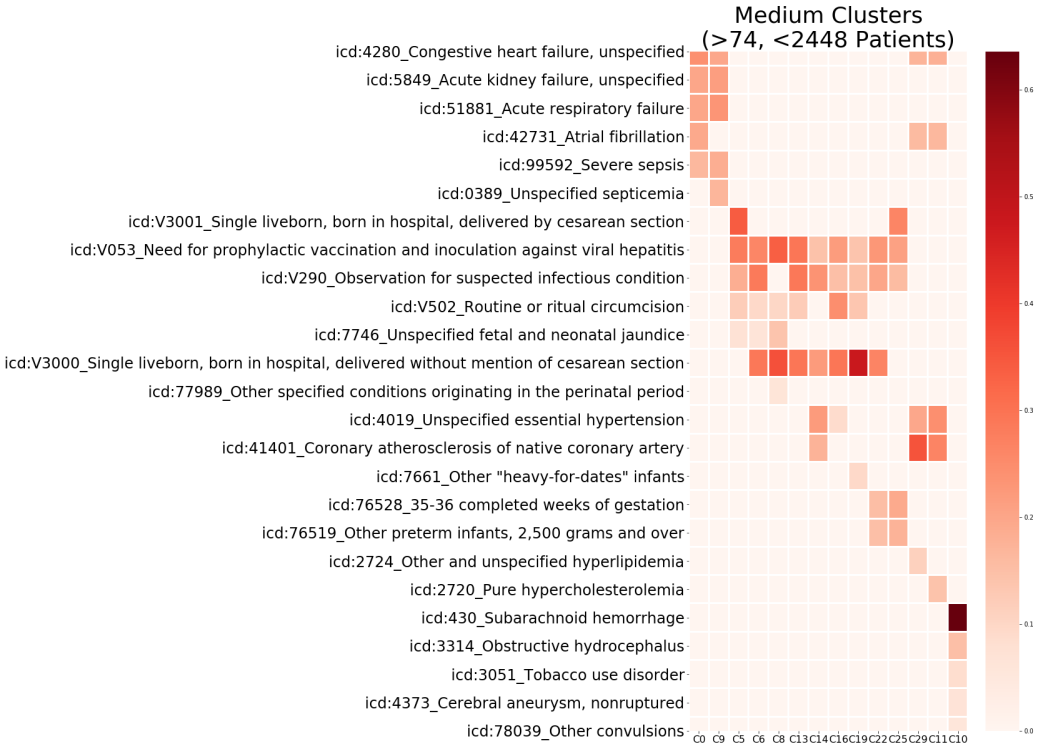
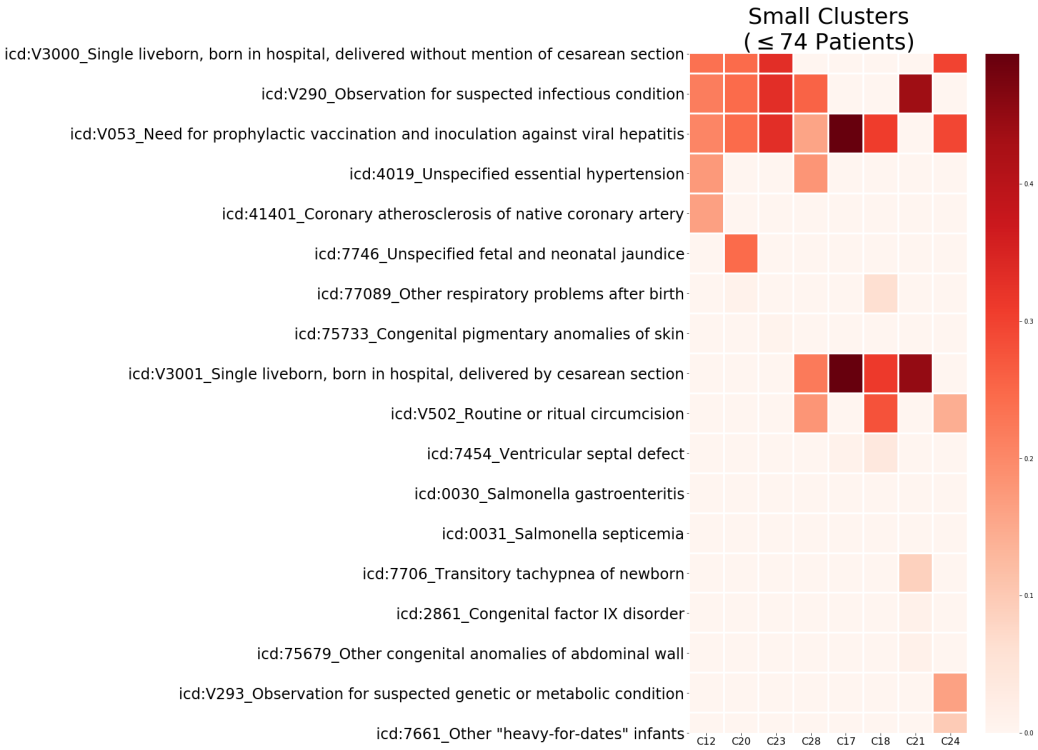


Figure 3-6: Top 5 ICD-9 codes for Small Clusters



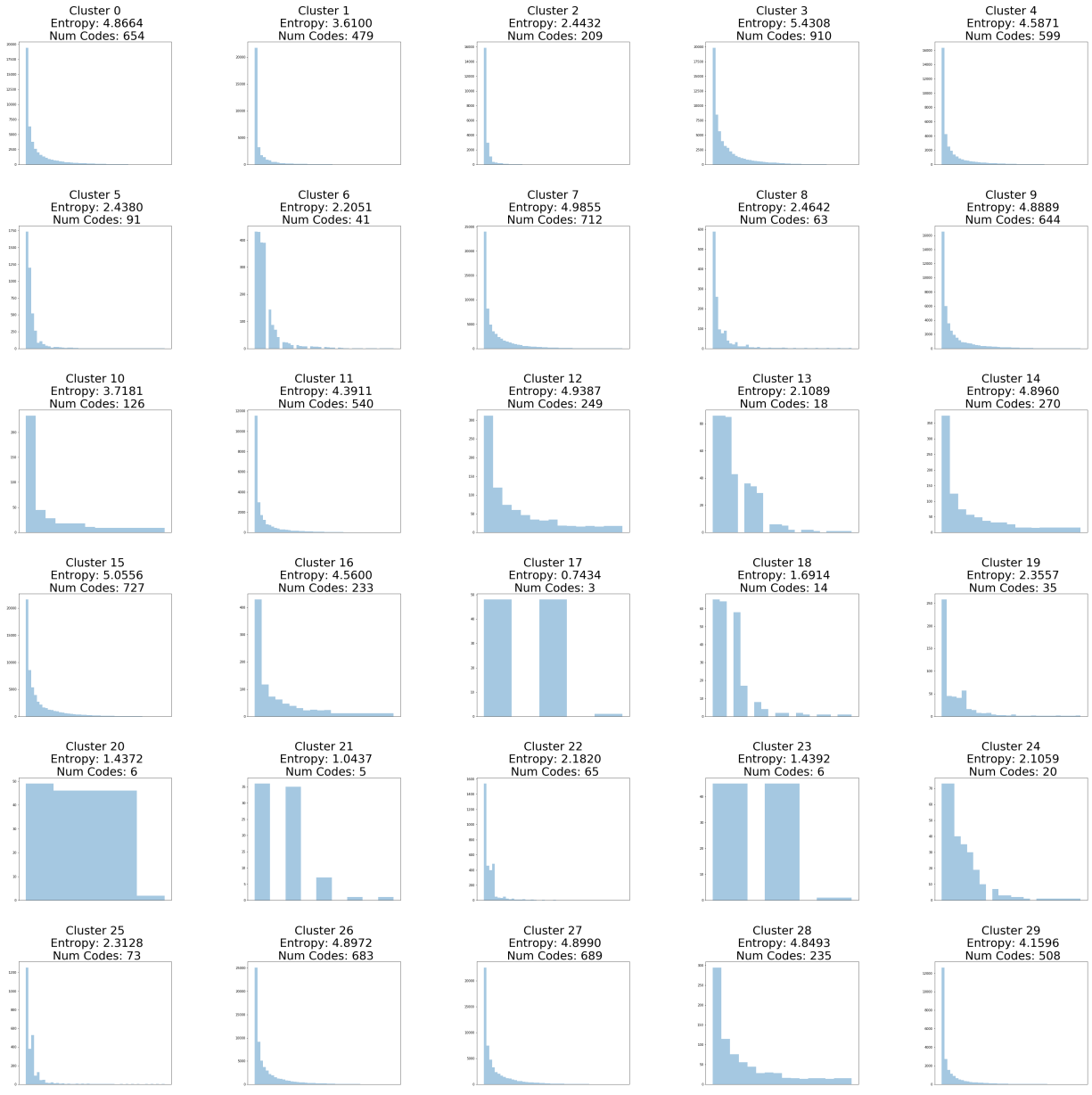
3.3.2 High-level analysis of all clusters

Hybrid Phecode-ICD distributions

Examining the contents of our clusters, we find that our method captures symptoms clusters as well as some underlying relationships. We first see this in the sorted distribution plot of Phecode-ICD features in Figure 3-7a (data summarized in Tables A.1 and A.2). Figure 3-7a contains 30 subplots corresponding to our 30 clusters and each subplot contains the sorted distribution of features as encoded by our Phecode-ICD scheme. Note that each subplot contains a *sorted* distribution, meaning that the histogram locations are not necessarily consistent across subplots (ex. the highest peak in Cluster 0 does not necessarily correspond to the highest peak in Cluster 1). We see that most clusters contain a single prominent feature followed by a rapid drop off with the range for the number of features in a cluster being 3 to 910. Based on this observation that most clusters contained a single dominant feature, we created a similarity network and colored it based on the most prominent feature.

Figure 3-7: Phecode-ICD distribution for all 30 clusters.

(a) Each plot contains the sorted distribution of Phecode-ICD features for that particular cluster in order to provide a sense of the mix of patients in each cluster. Note that this means there is no consistent ordering for the peaks across clusters (ex. the highest peak of Cluster 0 and Cluster 1 do not necessarily correspond to the same Phecode-ICD feature.)

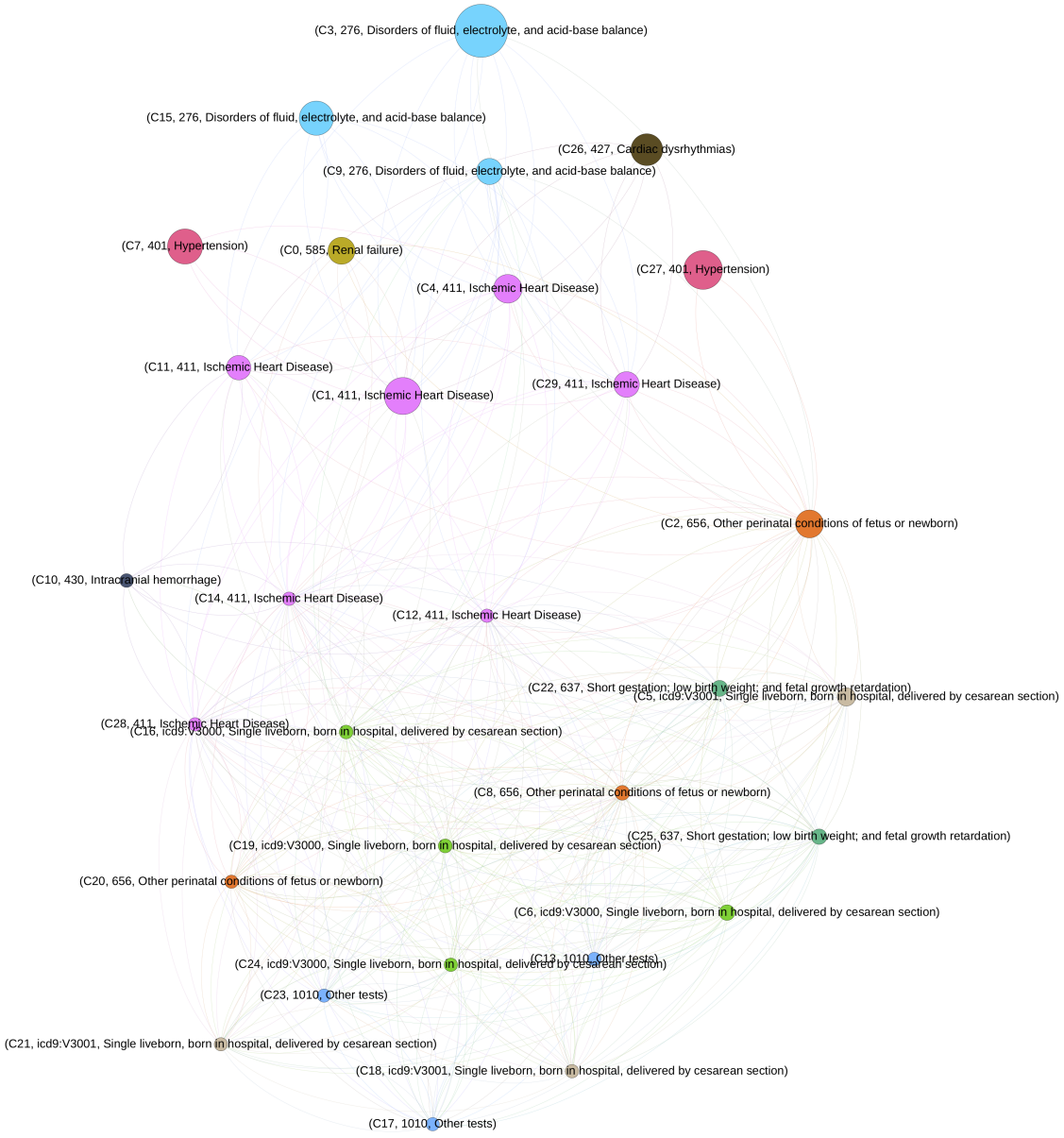


Weighted Jaccard similarity cluster network

Examining our cluster network in Figure 3-8a, we see that our weighted Jaccard similarity network identifies common conditions while also finding relations which potentially have clinical relevance. Here, cluster colors correspond to the most common code according to our hybrid Phecode-ICD features, cluster sizes correspond to the number of patients in the cluster, and edges have been pruned for visual clarity. We find that there are 11 unique Phecode-ICD features which belong to different families of symptoms (ex. "Intracranial hemorrhage" versus "Cardiac dysrhythmias") where the most common code is Phecode 411 — "Ischemic Heart Disease." Examining spatial relationships among clusters, we see that our network indeed groups similar symptom groups together. For example, we see near the top of the figure that clusters whose most common code is Phecode 276 — "Disorders of fluid, electrolyte, and acid-base balance" (shown in light blue) are located close to each other while the same is true at the bottom of the figure for some symptoms corresponding to infants (shown in green and grey). We also observe that there is a separation between the top and bottom portions of the network. This is likely due to the patient population in the MIMIC-III dataset which consists of adults and neonates. This is also supported by the fact that the conditions in the lower half of the graph correspond to child delivery and future works will be conducted to examine patient subpopulations more closely.

Figure 3-8: Cluster Network using weighted Jaccard similarity

(a) Clusters are colored according to most frequent Phecode-ICD feature in cluster with 13 colors in total. Cluster are labeled according to (Phecode-ICD feature, Phecode-ICD long-form name) where the prefix "icd9" indicates a ICD9 code and no prefix indicates a Phecode. Edges were pruned for visual clarity.



3.4 Discussion

For discussion purposes, we examine 3 clusters more closely and demonstrate that they capture symptoms that are clinically related. Here, we select clusters 3, 15, and 7 as these have the highest entropies based on their distributions of Phecode-ICD features. We next list their 5 most frequent Phecode-ICD features below:

- Cluster 3
    1. Phecode 276: "Disorders of fluid, electrolyte, and acid-base balance"
    2. Phecode 1008: "Crushing or internal injury to organs"
    3. Phecode 317: "Alcohol-related disorders"
    4. Phecode 285: "Other anemias"
    5. Phecode 819: "Skull and face fracture and other intercranial injury"
- Cluster 15
    1. Phecode 276: "Disorders of fluid, electrolyte, and acid-base balance"
    2. Phecode 585: "Renal failure"
    3. Phecode 509: "Respiratory failure, insufficiency, arrest"
    4. Phecode 250: "Diabetes mellitus"
    5. Phecode 038: "Septicemia"
- Cluster 7
    1. Phecode 401: "Hypertension"
    2. Phecode 427: "Cardiac dysrhythmias"
    3. Phecode 276: "Disorders of fluid, electrolyte, and acid-base balance"
    4. Phecode 250: "Diabetes mellitus"
    5. Phecode 428: "Congestive heart failure; nonhypertensive"

Examining the clusters, we see that we can associate the most frequent features with each other. For example, in Cluster 15, "Renal failure," "Septicemia," and "Diabetes mellitus" are known to be associated [49, 2]. These results indicate that clustering using EHR data alone is capable of highlighting high-level, clinically relevant associations between conditions and symptoms. Combining EHR data with genomic and other forms of data could yield even greater insights and improve healthcare treatments.

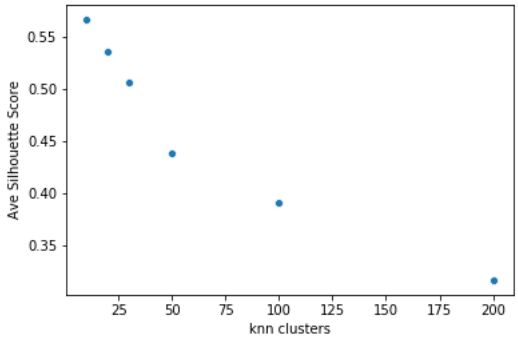
We obtained the results in this chapter using only ICD9 codes. While we did so due to the widespread prevalence of ICD9 codes in hospital datasets, we should consider other data types such as lab test results in future analysis. Additionally, to ensure our method generalizes, we should also assess these methods on other datasets.

3.4.1 Limitations

Selecting eigenvectors

Our selection of eigenvectors is based on empiricism rather than theory. While we have demonstrated that this method still yields promising results, our current lack of theoretical backing poses a potential challenge in generalizing our techniques.

Figure 3-9: Average silhouette scores for different number of K-Means clusters



Selecting number of K-Means clusters

Our empirical selection of the number of K-Means clusters was motivated by the need to obtain a qualitatively acceptable silhouette plot while also retaining a high level of detail in each cluster. For example, we tried various numbers of clusters, and their corresponding average silhouette scores are shown in Figure 3-9. While we achieved our highest silhouette score using 10 K-Means clusters, we ultimately chose to use 30 K-Means clusters as doing so yielded a similar score while providing more granular insights into patient symptoms.

Chapter 4

Variational Autoencoders (VAE)

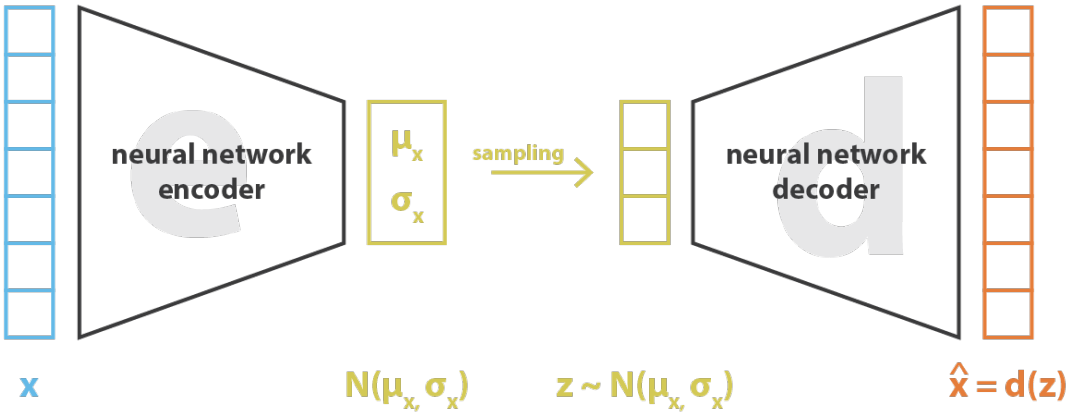
4.1 Background

A variational autoencoder (VAE) is a deep generative model that is capable of learning complicated distributions in an unsupervised manner. We take advantage of its ability to mapping complex data into lower-dimensions.

VAE learn to reconstruct input data and in the process are also able to learn high-level, generalizable patterns. They do so using two separate neural networks called encoder and decoder networks, as shown in Figure 4-1. The encoder maps input data into a much lower dimensional representation which is used to parameterize a continuous distribution or "latent space." In our case and with standard VAE, this distribution is is a isotopic Gaussian (ie. a multivariate Gaussian distribution with 0 mean and unit variance) [16, 22]. The decoder then samples a vector called the "latent representation" from the latent distribution and attempts to reconstruct the original input data.

We are particularly interested in the lower dimensional embedding vector, as it is a compact representation of the original data that is thought to preserve critical aspects of the original data while also reducing noise. This is because in order for a VAE to learn a lower dimensional embedding that is useful for reconstructing different data points, the encoder must learn compact features that are especially relevant for distinguishing one data point from another while discarding unimportant features.

Figure 4-1: Diagram of variational autoencoder from Towards Data Science [61]



Additionally, the dimension of the latent representation can be made smaller or larger to capture higher or lower level details. As a result, while a VAE outputs a reconstruction of our original data, the more important aspect that we leverage in this section is the encoder and its embedding vector.

4.1.1 Model

More formally, let x be an input datum and z be its latent representation. Through training, we aim to learn 1) an encoder for mapping input data x in discrete space to values z in continuous space and 2) a decoder for mapping values z in continuous space back to x in discrete space.

4.1.2 Loss Function

Because the encoder and decoder map their respective inputs onto different distributions — the encoder mapping to a parameterized, continuous distribution and the decoder mapping to the discrete distribution of the original input data — both optimize different loss functions. The encoder minimizes Kullback–Leibler (KL) divergence and the decoder minimizes reconstruction error. Here, we use binary cross entropy as our reconstruction error.

To examine the loss for a single input, let $x_i \in \mathbb{R}^m$ be a single input from $X =$

$\{x_1, ..., x_n\}$ and let x'_i be its final reconstruction. Additionally, to simplify our analysis without loss of generality, let all of X be used in a training batch. In our standard VAE model, we use the encoder to map our input x_i to its latent representation z_i, which is drawn from an isotropic Gaussian $N(0, I)$ where I is the identity matrix. An isotropic Gaussian $N(0, I)$ is parameterized by a mean and variance vector, so our encoder will map x_i to a corresponding mean vector μ_i and variance vector σ_i.

We can then express the loss $L(x_i)$ as a sum of KL-divergence $\mathrm{KL}(x_i)$ and binary cross entropy $\mathrm{BCE}(x_i, x'_i)$:

$$\begin{aligned} L(x_i) &= \mathrm{KL}(x_i) + \mathrm{BCE}(x_i, x'_i) \\ \mathrm{KL}(x_i) &= \frac{1}{2}\sum_{k=1}^{m}\left(1 + \log\left(\sigma_{i,k}\right) - \mu_{i,k}^2 - \sigma_{i,k}\right) \\ \mathrm{BCE}(x_i, x'_i) &= \frac{1}{m}\sum_{k=1}^{m}\left(-x_{i,k}\log(x'_{i,k}) - (1 - x_{i,k})\log(1 - x'_{i,k})\right) \end{aligned}$$

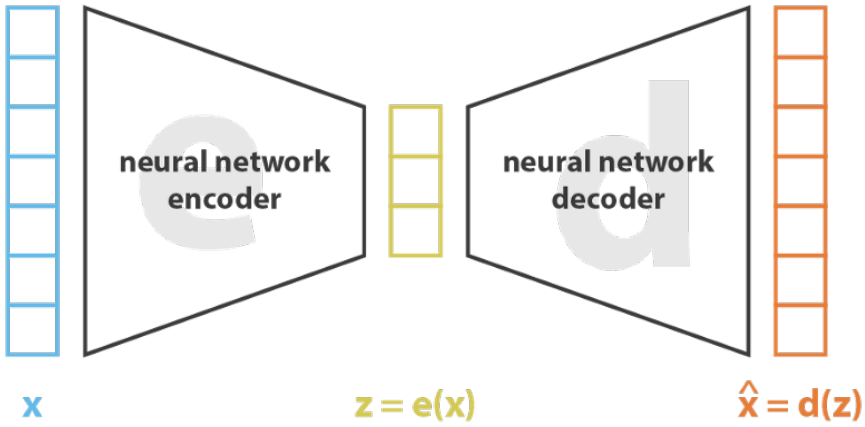
where we make use of the "reparameterization trick" to obtain the KL-divergence loss term [35]. We can then optimize this differentiable loss function using gradient descent.

It is important to note that our VAE maps data to $N(0, I)$ *in aggregate* over the data. In other words, while all data points will in aggregate map to a mean vector 0 and variance matrix I, each individual data point x_i may nontrivially map to a mean vector that is not necessarily 0 and a variance vector that is not necessarily a column in I.

4.1.3 VAE versus Autoencoders (AEs)

It's worth briefly exploring why VAE project data onto a distribution by examining its close relative, autoencoders (AE). AE are similar to VAE — they are neural network-based models that are used to reconstruct data and learn high-level features, and they are also composed of an encoder and decoder network. However, the main architectural difference is that the encoder of an AE does not project data onto a dis-

Figure 4-2: Diagram of autoencoder from Towards Data Science [61]



tribution; instead, it reduces the dimensionality of the input data by "bottlenecking" the data to a lower dimensional representation. This is shown in Figure 4-2 where the encoder of an AE creates a lower-dimensional vector embedding that is directly used by the decoder, whereas the VAE encoder uses the embedding to approximate a distribution.

The VAE's use of a distribution has several added benefits that the AE does not, such as creating an embedding with disentangled representations [26] and generating synthetic data [9]. The choice of distribution can also be modified depending on the specific project or desired output. However, detailed discussions of these topics is beyond the scope of this thesis.

4.2 Experiment

While we experimented with several architectures, we were unable to obtain efficacious results. As a result, instead of enumerating results, we present the experiment which motivated our switching to spectral clustering.

In these preliminary experiments, our primary goal was to automatically learn compact patient representations that could be clustered into distinct groups and then visualized. We also aimed to produce interpretable results as doing so is especially

relevant to healthcare. However, due to insufficient results we forego cluster analysis and instead highlight the insights obtained from analyzing our visualizations.

4.2.1 Data Preprocessing

Just as in Section 3.2.1, we use ICD-9 diagnostic codes to compute a binary 0-1 matrix where each patient is represented by a binary vector. In this binary vector, a 1 represents the presence an ICD-9 diagnostic code. Doing so results in a matrix of shape 46,520 patients $\times$ 6,984 ICD-9 diagnostic codes. See Section 3.2.1 for more details.

4.2.2 Model Architecture and Training

To capture high-level features and ensure interpretable outputs, we use a VAE whose encoder is a compact neural network with layers containing 250, 500, 250, 100, and 25 nodes respectively and whose decoder consists of a single layer with 6,984 nodes. Note this means our latent representation has 25 dimensions. We choose to use a neural network-based model in this setting as our data are high dimensional and there could be non-linear relationships between features. Additionally, we choose to use a single-layer decoder as doing so theoretically allows us to directly map our embedding vector to final output activations, although this ultimately was not the case. We trained our model for 55 epochs with a learning rate of 0.001 and a batch size of 32.

4.2.3 Visualization

Uniform Manifold Approximation and Projection for Dimension Reduction (UMAP)

We use Uniform Manifold Approximation and Projection for Dimension Reduction or UMAP on our latent representations [43] followed by coloring based on manual keyword selection. More specifically, we obtain a 25-dimensional representation for each patient, use UMAP to obtain 2- and 3- dimensional embeddings, and then plot the embeddings.

Keyword Selection

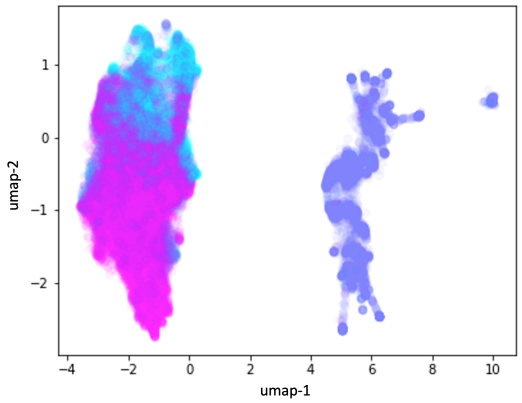
To find keywords, we inspect the UMAP plots and manually select patients from regions that appear to be clustered. Then, we extract a subset of those patients' ICD-9 codes and use their corresponding titles to obtain keywords. Finally, we then find all patients whose ICD-9 codes have any matching keywords and then color them accordingly. For example, lets assume the manually selected patient had the ICD-9 code V3000 corresponding to "Single liveborn, born in hospital, delivered by cesarean section." From this title, we would then manually extract keywords such as "liveborn" and "born," and then lookup all ICD-9 codes whose title contains those keywords. This lookup would yield other related ICD-9 codes, such as V30001. After obtaining all of these ICD-9 codes whose titles have matching keywords, we then find all patients with those corresponding ICD-9 codes and color them.

Ultimately, we arrived at keywords pertaining to neonates and hearts, where our neonate-related keywords were "congenital," "infant," "newborn," "neonatal," "born," and "birth" and our heart-related keywords were "heart," "atrial," "coronary," "hypertension," and "vascular." If a patient contained both neonates and heart keywords, then it was assigned the neonate color.

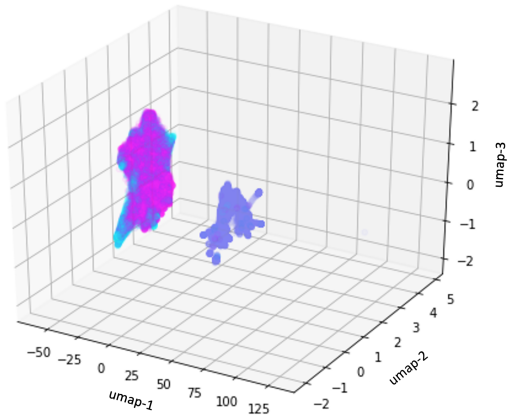
We recognize that this method is rather naive. For example, it ignores the fact that some patients have multiple ICD-9 codes or that some patients may not be accurately represented by whatever keywords or ICD-9 codes we obtain. However, this method was still valuable in helping us understand the shortcomings in our methodology.

4.3 Results

Figure 4-3: UMAP plots for 25-dimensional VAE latent representations.



(a) 2D UMAP. Light blue is the default color, pink denotes patients with heart keywords, and purple denotes patients with baby keywords.



(b) 3D UMAP. Light blue is the default color, pink denotes patients with heart keywords, and purple denotes patients with baby keywords.

Our UMAP plots reveal a separation between patients with neonate-related keywords and other patients(Figure 4-3). At the same time, we do not observe any meaningful separation between patients with heart-related keywords and other patients.

The separation of neonate-related keywords and other patients may likely be due to other factors and data that we did not use in these experiments, such as the age of the patient. Indeed, this is supported by the fact that we achieved similar results for other categories of keywords. For example, selecting kidney-related keywords (results not shown) these yielded a similar coloring. These preliminary results then indicate that our naive VAE method may require additional data in order to identify patterns.

Based on these exploratory VAE experiments, we realized we had to adopt a more hands-on and deterministic approach for generating patient representations in order to obtain more granular separation between patients. As a result, we adopted a spectral clustering-based approach.

Chapter 5

Conclusion

In this thesis, we conducted two sets of experiments, one with modified spectral clustering and another with variational autoencoders, using International Classification of Disease (ICD)-9 codes from the Medical Information Mart for Intensive Care (MIMIC)-III dataset to obtain compact patient representations. In our spectral clustering experiments, we clustered these patient representations and then conducted cluster-level analysis by examining the top 5 most frequent symptoms within each cluster as well as higher-level analysis by constructing a cluster network using weighted Jaccard similarity. Through examining a subset of individual clusters and the most frequently occurring symptoms within those clusters, we find that our clusters capture conditions and symptoms that are clinically associated. Through examining our cluster network, we find that our method captures high-level patterns where clusters with similar most-frequent symptoms are also more closely located to each other. In our variational autoencoder experiments, we found that visualizations of our patient representations showed clear separation between patients with conditions containing a subset of neonate-related keywords and other conditions, but that ultimately additional research and data types are likely needed to improve the quality of the patient representations.

Appendix A

Tables

Table A.1: Top Phecode-ICD9 Cluster Summary

Top Code	Name	# Clusters
Phecode 411	Ischemic Heart Disease	7
ICD9 V3000	Single liveborn, born in hospital, delivered by cesarean section	4
ICD9 V3001	Single liveborn, born in hospital, delivered by cesarean section	3
Phecode 276	Disorders of fluid, electrolyte, and acid-base balance	3
Phecode 656	Other perinatal conditions of fetus or newborn	3
Phecode 1010	Other tests	3
Phecode 637	Short gestation; low birth weight; and fetal growth retardation	2
Phecode 401	Hypertension	2
Phecode 430	Intracranial hemorrhage	1
Phecode 427	Cardiac dysrhythmias	1
Phecode 585	Renal failure	1

Table A.2: Size and most common Phecode-ICD9 code for each cluster

Cluster	Size	Total Codes	Most Common Code	Condition Name
0	2340	654	Phecode 585	Renal Failure
1	4013	479	Phecode 411	Ischemic Heart Disease
2	2484	209	Phecode 656	Other perinatal conditions of fetus or newborn
3	6687	910	Phecode 276	Disorders of fluid, electrolyte, and acid-base balance
4	2604	599	Phecode 411	Ischemic Heart Disease
5	883	91	ICD9 V3001	Single liveborn, born in hospital, delivered by cesarean section
6	435	41	ICD9 V3000	Single liveborn, born in hospital, delivered by cesarean section
7	3693	712	Phecode 401	Hypertension
8	291	63	Phecode 656	Other perinatal conditions of fetus or newborn
9	2187	644	Phecode 276	Disorders of fluid, electrolyte, and acid-base balance
10	119	126	Phecode 430	Intracranial hemorrhage
11	1914	540	Phecode 411	Ischemic Heart Disease
12	56	249	Phecode 411	Ischemic Heart Disease
13	86	18	Phecode 1010	Other tests
14	78	270	Phecode 411	Ischemic Heart Disease
15	3545	727	Phecode 276	Disorders of fluid, electrolyte, and acid-base balance
16	121	233	ICD9 V3000	Single liveborn, born in hospital, delivered by cesarean section
17	48	3	Phecode 1010	Other tests
18	65	14	ICD9 V3001	Single liveborn, born in hospital, delivered by cesarean section
19	146	35	ICD9 V3000	Single liveborn, born in hospital, delivered by cesarean section
20	46	6	Phecode 656	Other perinatal conditions of fetus or newborn
21	36	5	ICD9 V3001	Single liveborn, born in hospital, delivered by cesarean section
22	498	65	Phecode 637	Short gestation; low birth weight; and fetal growth retardation
23	45	6	Phecode 1010	Other tests
24	73	20	ICD9 V3000	Single liveborn, born in hospital, delivered by cesarean section
25	393	73	Phecode 637	Short gestation; low birth weight; and fetal growth retardation
26	3152	683	Phecode 427	Cardiac dysrhythmias
27	4313	689	Phecode 401	Hypertension
28	65	235	Phecode 411	Ischemic Heart Disease
29	2101	508	Phecode 411	Ischemic Heart Disease

Bibliography

[1] What is an electronic health record (ehr)?, Sep 2019.

[2] Kevin C Abbott, Malcolm G Napier, and Lawrence Y Agodoa. Hospitalizations for bacterial septicemia in patients with end stage renal disease due to diabetes on the renal transplant waiting list. *Journal of nephrology*, 15(3):248–254, 2002.

[3] Julia Adler-Milstein and Ashish K Jha. Hitech act drove large gains in hospital electronic health record adoption. *Health Affairs*, 36(8):1416–1422, 2017.

[4] Jeffrey J Bazarian, Peter Veazie, Sohug Mookerjee, and E Brooke Lerner. Accuracy of mild traumatic brain injury case ascertainment using icd-9 codes. *Academic emergency medicine*, 13(1):31–38, 2006.

[5] Catherine M Bender, Fisun Süenuzun Ergÿn, Margaret Q Rosenzweig, Susan M Cohen, and Susan M Sereika. Symptom clusters in breast cancer across 3 phases of the disease. *Cancer Nursing*, 28(3):219–225, 2005.

[6] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[7] Elena Birman-Deych, Amy D Waterman, Yan Yan, David S Nilasena, Martha J Radford, and Brian F Gage. Accuracy of icd-9-cm codes for identifying cardiovascular and stroke risk factors. *Medical care*, pages 480–485, 2005.

[8] Kathy L Brouch. Where in the world is icd-10? *Where in the World Is ICD-10?/AHIMA, American Health Information Management Association*, 2000.

[9] Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*, 2015.

[10] Douglas Carroll, Anna C Phillips, Catharine R Gale, and G David Batty. Generalized anxiety and major depressive disorders, their comorbidity and hypertension in middle-aged men. *Psychosomatic Medicine*, 72(1):16–19, 2010.

[11] Moses S Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 380–388, 2002.

[12] Travers Ching, Daniel S Himmelstein, Brett K Beaulieu-Jones, Alexandr A Kalinin, Brian T Do, Gregory P Way, Enrico Ferrero, Paul-Michael Agapow, Michael Zietz, Michael M Hoffman, et al. Opportunities and obstacles for deep learning in biology and medicine. *Journal of The Royal Society Interface*, 15(141):20170387, 2018.

[13] Fan RK Chung and Fan Chung Graham. *Spectral graph theory*. Number 92. American Mathematical Soc., 1997.

[14] Joshua C Denny, Lisa Bastarache, Marylyn D Ritchie, Robert J Carroll, Raquel Zink, Jonathan D Mosley, Julie R Field, Jill M Pulley, Andrea H Ramirez, Erica Bowton, et al. Systematic comparison of phenome-wide association study of electronic medical record data and genome-wide association study data. *Nature biotechnology*, 31(12):1102, 2013.

[15] Marylin J Dodd, Christine Miaskowski, and Steven M Paul. Symptom clusters and their effect on the functional status of patients with cancer. In *Oncology nursing forum*, volume 28, 2001.

[16] Carl Doersch. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*, 2016.

[17] Jeremy U Espino and Michael M Wagner. Accuracy of icd-9-coded chief complaints and diagnoses for the detection of acute respiratory illness. In *Proceedings of the AMIA Symposium*, page 164. American Medical Informatics Association, 2001.

[18] Centers for Disease Control, Prevention, et al. International classification of diseases, ninth revision, clinical modification (icd-9-cm), 2013.

[19] Audrey G Gift, Anita Jablonski, Manfred Stommel, C William Given, et al. Symptom clusters in elderly patients with lung cancer. In *Oncology nursing forum*, volume 31, pages 203–218. ONCOLOGY NURSING SOCIETY, 2004.

[20] Kwang-Il Goh, Michael E Cusick, David Valle, Barton Childs, Marc Vidal, and Albert-László Barabási. The human disease network. *Proceedings of the National Academy of Sciences*, 104(21):8685–8690, 2007.

[21] Larry B Goldstein. Accuracy of icd-9-cm coding for the identification of patients with acute ischemic stroke: effect of modifier codes. *Stroke*, 29(8):1602–1604, 1998.

[22] Ian Goodfellow. Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2016.

[23] Mengfei Guo, Yanan Yu, Tiancai Wen, Xiaoping Zhang, Baoyan Liu, Jin Zhang, Runshun Zhang, Yanning Zhang, and Xuezhong Zhou. Analysis of disease comorbidity patterns in a large-scale china population. *BMC Medical Genomics*, 12(12):177, 2019.

[24] Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.

[25] Anita C Hazelwood. Icd-9 cm to icd-10 cm: implementation issues and challenges. *ICD-9 CM to ICD-10 CM: Implementation Issues and Challenges/AHIMA, American Health Information Management Association*, 2003.

[26] Irina Higgins, Loic Matthey, Xavier Glorot, Arka Pal, Benigno Uria, Charles Blundell, Shakir Mohamed, and Alexander Lerchner. Early visual concept learning with unsupervised deep learning. *arXiv preprint arXiv:1606.05579*, 2016.

[27] Suzanne C Ho, Sieu Gaen Chan, Yin Bing Yip, Anna Cheng, Qilong Yi, and Cynthia Chan. Menopausal symptoms and symptom clustering in chinese women. *Maturitas*, 33(3):219–227, 1999.

[28] Jin Huang, Feiping Nie, and Heng Huang. Spectral rotation versus k-means in spectral clustering. In *Twenty-Seventh AAAI Conference on Artificial Intelligence*, 2013.

[29] Li Huang, Yifeng Yin, Zeng Fu, Shifa Zhang, Hao Deng, and Dianbo Liu. Loadaboost: Loss-based adaboost federated machine learning on medical data. *arXiv preprint arXiv:1811.12629*, 2018.

[30] Peter B Jensen, Lars J Jensen, and Søren Brunak. Mining electronic health records: towards better research applications and clinical care. *Nature Reviews Genetics*, 13(6):395–405, 2012.

[31] Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3:160035, 2016.

[32] Ravi Kannan, Santosh Vempala, and Adrian Vetta. On clusterings: Good, bad and spectral. *Journal of the ACM (JACM)*, 51(3):497–515, 2004.

[33] Jeffrey L Kibler, Kavita Joshi, and Mindy Ma. Hypertension in relation to posttraumatic stress disorder and depression in the us national comorbidity survey. *Behavioral Medicine*, 34(4):125–132, 2009.

[34] Hee-Ju Kim, Deborah B McGuire, Lorraine Tulman, and Andrea M Barsevick. Symptom clusters: concept analysis and clinical implications for cancer nursing. *Cancer nursing*, 28(4):270–282, 2005.

[35] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[36] Barry Knishkowy, Hava Palti, Charles Tima, Bella Adler, and Rosa Gofin. Symptom clusters among young adolescents. *Adolescence*, 30(118):351, 1995.

[37] Junghye Lee, Jimeng Sun, Fei Wang, Shuang Wang, Chi-Hyuck Jun, and Xiaoqian Jiang. Privacy-preserving patient similarity learning in a federated environment: development and analysis. *JMIR medical informatics*, 6(2):e20, 2018.

[38] Yue Li and Manolis Kellis. A latent topic model for mining heterogenous non-randomly missing electronic health records data. *arXiv preprint arXiv:1811.00464*, 2018.

[39] Edo Liberty, Franco Woolfe, Per-Gunnar Martinsson, Vladimir Rokhlin, and Mark Tygert. Randomized algorithms for the low-rank approximation of matrices. *Proceedings of the National Academy of Sciences*, 104(51):20167–20172, 2007.

[40] Dianbo Liu, Dmitriy Dligach, and Timothy Miller. Two-stage federated phenotyping and patient representation learning. *arXiv preprint arXiv:1908.05596*, 2019.

[41] Xinrui Lyu, Matthias Hueser, Stephanie L Hyland, George Zerveas, and Gunnar Rätsch. Improving clinical predictions through unsupervised time series representation learning. *arXiv preprint arXiv:1812.00490*, 2018.

[42] Tengfei Ma, Cao Xiao, and Fei Wang. Health-atm: A deep architecture for multifaceted patient health record representation and risk prediction. In *Proceedings of the 2018 SIAM International Conference on Data Mining*, pages 261–269. SIAM, 2018.

[43] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[44] Riccardo Miotto, Li Li, Brian A Kidd, and Joel T Dudley. Deep patient: an unsupervised representation to predict the future of patients from the electronic health records. *Scientific reports*, 6(1):1–10, 2016.

[45] Riccardo Miotto, Fei Wang, Shuang Wang, Xiaoqian Jiang, and Joel T Dudley. Deep learning for healthcare: review, opportunities and challenges. *Briefings in bioinformatics*, 19(6):1236–1246, 2018.

[46] Andrew Y Ng, Michael I Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in neural information processing systems*, pages 849–856, 2002.

[47] Marzia Polito and Pietro Perona. Grouping and dimensionality reduction by locally linear embedding. In *Advances in neural information processing systems*, pages 1255–1262, 2002.

[48] Tom J Pollard, Alistair EW Johnson, Jesse D Raffa, Leo A Celi, Roger G Mark, and Omar Badawi. The eicu collaborative research database, a freely available multi-center database for critical care research. *Scientific data*, 5:180178, 2018.

[49] Neil R Powe, Bernard Jaar, Susan L Furth, Judith Hermann, and William Briggs. Septicemia in dialysis patients: incidence, risk factors, and prognosis. *Kidney international*, 55(3):1081–1090, 1999.

[50] Gang Qian, Shamik Sural, Yuelong Gu, and Sakti Pramanik. Similarity between euclidean and cosine angle distance for nearest neighbor queries. In *Proceedings of the 2004 ACM symposium on Applied computing*, pages 1232–1237, 2004.

[51] Daniele Ravì, Charence Wong, Fani Deligianni, Melissa Berthelot, Javier Andreu-Perez, Benny Lo, and Guang-Zhong Yang. Deep learning for health informatics. *IEEE journal of biomedical and health informatics*, 21(1):4–21, 2016.

[52] Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

[53] Steven H Sanders and Steven F Brena. Empirically derived chronic pain patient subgroups: the utility of multidimensional clustering to identify differential treatment effects. *Pain*, 54(1):51–56, 1993.

[54] DB Scheurer, LS Hicks, EF Cook, and JL Schnipper. Accuracy of icd-9 coding for clostridium difficile infections: a retrospective cohort. *Epidemiology & Infection*, 135(6):1010–1013, 2007.

[55] Daniel A Spielman and Shang-Hua Teng. Spectral partitioning works: Planar graphs and finite element meshes. In *Proceedings of 37th Conference on Foundations of Computer Science*, pages 96–105. IEEE, 1996.

[56] X Yu Stella and Jianbo Shi. Multiclass spectral clustering. In *null*, page 313. IEEE, 2003.

[57] Cathie Sudlow, John Gallacher, Naomi Allen, Valerie Beral, Paul Burton, John Danesh, Paul Downey, Paul Elliott, Jane Green, Martin Landray, et al. Uk biobank: an open access resource for identifying the causes of a wide range of complex diseases of middle and old age. *PLoS medicine*, 12(3), 2015.

[58] Qiuling Suo, Fenglong Ma, Ye Yuan, Mengdi Huai, Weida Zhong, Jing Gao, and Aidong Zhang. Deep patient similarity learning for personalized healthcare. *IEEE transactions on nanobioscience*, 17(3):219–227, 2018.

[59] Arthur Szlam, Yuval Kluger, and Mark Tygert. An implementation of a randomized algorithm for principal component analysis. *arXiv preprint arXiv:1412.3510*, 2014.

[60] Maxim Topaz, Leah Shafran-Topaz, and Kathryn H Bowles. Icd-9 to icd-10: evolution, revolution, and current debates in the united states. *Perspectives in Health Information Management/AHIMA, American Health Information Management Association*, 10(Spring), 2013.

[61] Towards Data Science. Understanding variational autoencoders (vaes), 2019.

[62] Jaw-Shiun Tsai, Chih-Hsun Wu, Tai-Yuan Chiu, and Ching-Yu Chen. Significance of symptom clustering in palliative care of advanced cancer patients. *Journal of pain and symptom management*, 39(4):655–662, 2010.

[63] Barbara J Turner, Christopher S Hollenbeak, Mark Weiner, Thomas Ten Have, and Simon SK Tang. Effect of unrelated comorbid conditions on hypertension management. *Annals of internal medicine*, 148(8):578–586, 2008.

[64] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

[65] Declan Walsh and Lisa Rybicki. Symptom clustering in advanced cancer. *Supportive care in cancer*, 14(8):831–836, 2006.

[66] Liang Wang, Tieniu Tan, Huazhong Ning, and Weiming Hu. Silhouette analysis-based gait recognition for human identification. *IEEE transactions on pattern analysis and machine intelligence*, 25(12):1505–1518, 2003.

[67] Wei-Qi Wei, Lisa A Bastarache, Robert J Carroll, Joy E Marlo, Travis J Osterman, Eric R Gamazon, Nancy J Cox, Dan M Roden, and Joshua C Denny. Evaluating phecodes, clinical classification software, and icd-9-cm codes for phenome-wide association studies in the electronic health record. *PloS one*, 12(7), 2017.

[68] Wei-Hung Weng and Peter Szolovits. Representation learning for electronic health records. *arXiv preprint arXiv:1909.09248*, 2019.

[69] Jionglin Wu, Jason Roy, and Walter F Stewart. Prediction modeling using ehr data: challenges, strategies, and a comparison of machine learning approaches. *Medical care*, pages S106–S113, 2010.

[70] Chao Zhao, Jingchi Jiang, Yi Guan, Xitong Guo, and Bin He. Emr-based medical knowledge representation and inference via markov random fields and distributed representation learning. *Artificial intelligence in medicine*, 87:49–59, 2018.