Bigeometric Organization of Deep Nets

Alexander Cloninger[1], Ronald R. Coifman[1], Nicholas Downing[2] and Harlan M. Krumholz[2]

[1]Applied Mathematics Program, Yale University
[2]Center for Outcomes Research and Evaluation, Yale University

Abstract

In this paper, we build an organization of high-dimensional datasets that cannot be cleanly embedded into a low-dimensional representation due to missing entries and a subset of the features being irrelevant to modeling functions of interest. Our algorithm begins by defining coarse neighborhoods of the points and defining an expected empirical function value on these neighborhoods. We then generate new non-linear features with deep net representations tuned to model the approximate function, and re-organize the geometry of the points with respect to the new representation. Finally, the points are locally z-scored to create an intrinsic geometric organization which is independent of the parameters of the deep net, a geometry designed to assure smoothness with respect to the empirical function. We examine this approach on data from the Center for Medicare and Medicaid Services Hospital Quality Initiative, and generate an intrinsic low-dimensional organization of the hospitals that is smooth with respect to an expert driven function of quality.

1 Introduction

Finding low dimensional embeddings of high dimensional data is vital in understanding the organization of unsupervised data sets. However, most embedding techniques rely on the assumption that the data set is locally Euclidean [7, 15, 1]. In the case that features carry implicit weighting, some features are possibly irrelevant, and most points are missing some subset of the features, Euclidean neighborhoods can become spurious and lead to poor low dimensional representations.

In this paper, we develop the method of expert driven functional discovery to deal with the issue of spurious neighborhoods in data sets with high dimensional contrasting features. This allows small amounts of input and ranking from experts to propagate through the data set in a non linear, smooth fashion. We then build a distance metric based off these opinions that learns the invariant and irrelevant features from this expert driven function.

Finally, we locally normalize this distance metric to generate a global embedding of the data into a homogeneous space.

An example to keep in mind throughout the paper, an idea we expand upon in Section 4, is a data set containing publicly-reported measurements of hospital quality. The Center for Medicare and Medicaid Services Hospital Quality Initiative reports approximately 100 different measures describing various components of the quality of care provided at Medicare-Certified hospitals across the United States. These features range in measuring hospital processes, patient experience, safety, rates of surgical complications, and rates of various types of readmission and mortality. There are more than $5,000$ hospitals that reported at least on measure during 2014, but only $1,614$ hospitals with 90% measures reported. The measures are computed quarterly, and are publicly available through the Hospital Compare website [8]. The high dimensional nature of these varied measures make comprehensive inferences about hospital quality impossible without summarizing statistics.

Our goal is more than just learning a ranking function f on the set of hospitals X. We are trying to characterize the cohort of hospitals and organize the geometry of the data set, and learn a multi-dimensional embedding of the data for which the ranking function is smooth. This gives an understanding of the data that doesn't exist with a one dimensional ranking function. Specifically, we are looking for meta-features of the data in order to build a metric $\rho : X \times X \to \mathbb{R}^+$ that induces a small Lipschitz constant on the function f, as well as on features measured by CMS.

An example of this organization is shown in Figure 1. The organization is generated via our algorithm of expert driven functional discovery, the details of which are found in Sections 2 and 3. The colors in each image correspond to three notable CMS features: risk standardized 30 day hospital-wide readmission, patient overall rating of the hospital, and risk standardized 30 day mortality for heart failure. This organization successfully separates hospitals into regimes for which each feature is relatively smooth.

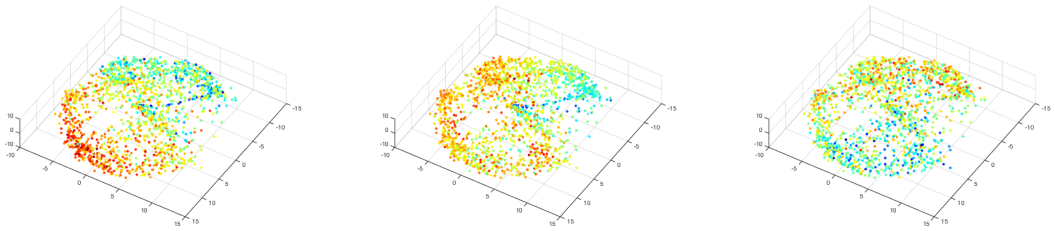

Figure 1: Organization colored by: (left) risk standardized 30 day hospital wide readmission, (center) percent patients rating overall hospital 9 or 10 out of 10, (right) risk standardized 30 day mortality for heart failure. Embedding generated via bigeometric organization of deep nets. Red is good performance, blue is bad.

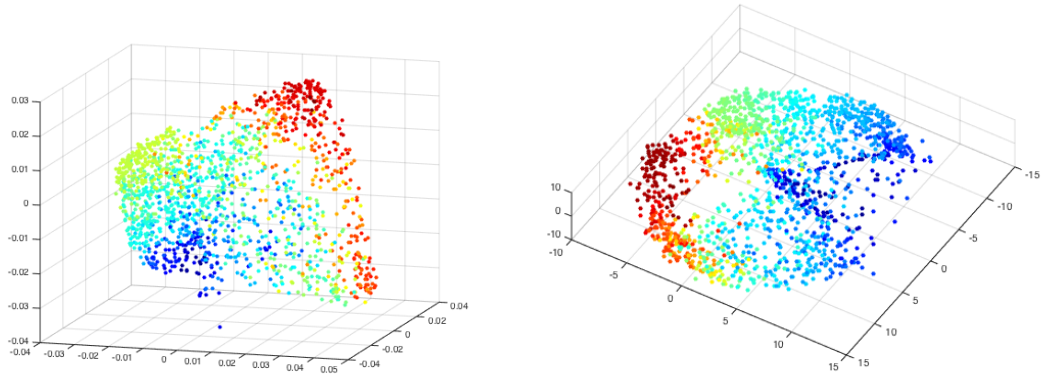

Figure 2: (left) unnormalized embedding, (right) z-scored embedding.

Our organization is accomplished via a two step processing of the data. First, we look to characterize the topology of the data set by creating a stable representation of X such that neighborhoods can be differentiated at varying scales, and neighborhoods are preserved across varying parameters of the algorithm. This is done in Sections 2 and 3, using a spin cycling of deep nets trained on some expert driven function. Second, we build a metic on that topology by taking a local Mahalanobis distance on the stable neighborhoods (i.e. local z-scoring) [11, 16]. This is done in Section 3.3, and guarantees that the induced metric is homogeneous. This means that, if U_x denotes the neighborhood of a point x, then $\rho(x, y)$ for $x, y \in U_x$ measures the same notion of distance as $\rho(x', y')$ for $x', y' \in U_{x'}$.

Figure 2 demonstrates the issue we refer to here. The left image shows an organization of the hospitals in which neighborhoods have not been normalized. The red points refer to "good" hospitals, and the blue refer to "bad" hospitals (in a naive sense). The organization on the left gives no notion of the spread of the points and leaves certain questions unanswerable (i.e. are "good" hospitals as diverse as "bad" hospitals). However, if we locally z-score the regions of the hospitals, as in the right figure, it the global notion of distance is normalized and it becomes clear that "good" hospitals share many more similarities amongst themselves than "bad" hospitals do among themselves.

Also, by taking a local z-scoring of the features, we generate an organization that is dependent only on the neighborhoods U_x, rather than being dependent on the specific representations used. Figure 3 shows the organizations of the hospitals generated by algorithms with two very different parameter sets which, after z-scoring the neighborhoods, generate similar embeddings. More details about this embedding can be found in Section 4.3.3.

However, discovering the topology of the hospitals is non-trivial. The features may have significant disagreement, and not be strongly correlated across the population. To examine

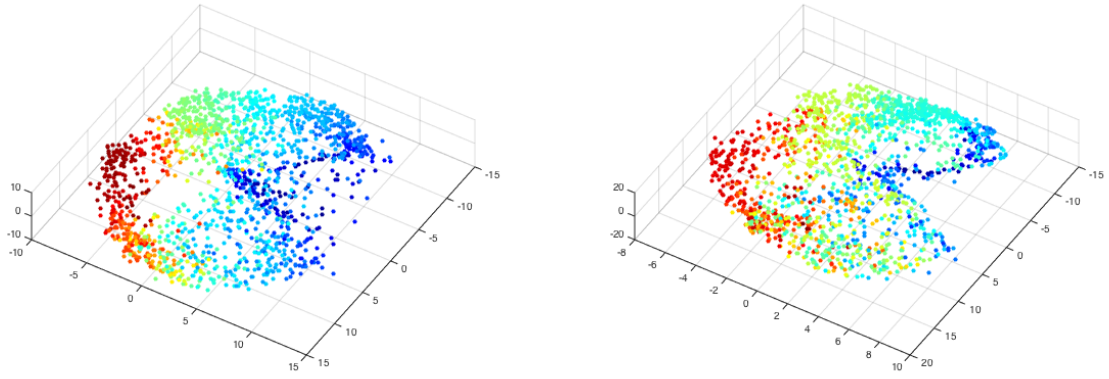

Figure 3: Two sets of organizations with similar neighborhood structure. The representations used to generate these embeddings are fundamentally different, with the right figure using 5× as many features as the left, and with features being generated with vastly different algorithmic parameters.

these relationships, one can consider linear correlations via principal component analysis. The eigenvalues of the correlation matrix do not show the characteristic drop off shown in linear low dimensional data sets. In fact, 76 of the 86 eigenvalues are above 1% the size of the largest eigenvalue. Previous medical literature has also detailed the fact that many of the features don't always correlate [10, 3].

For this reason, there does not exist an organization for which all features are smooth and monotonically increasing. This is why the meta-features, and organization, must be driven by minimal external expert opinion. This observation makes the goal of our approach three fold: develop an organization of the data that is smooth with respect to as many features as possible, build a ranking function f that agrees with this organization, and minimize the amount of external input necessary to drive the system.

Our expert driven functional discovery algorithm blends the data set organization of diffusion maps and coupled partition trees with the rich set of non-linear features generated by deep learning and stacked neural nets. We build an initial organization of the data via coupled partition trees [5], and use this partitioning to generate pseudopoints that accurately represent the span of the data at a coarse scale. This step is explained in Section 2. This organization is analyzed and used as an input to a series of stacked neural nets, which learn invariant representations of the data and separate disparate clusters of points. This step is explained in Section 3. See [2] for a review of stacked neural nets and deep learning.

It is important to note that our use of stacked neural nets is different from traditional deep learning applications. We discuss these differences in Section 3.5. The purpose of

using deep learning and organizing of the generated representations is to create a notion of fine neighborhoods between points; neighborhoods where the number of neighbors scales smoothly with the distance metric.

We then examine and validate our algorithm on the CMS Quality Initiative features in Section 4.

2 Information Organization and Expert Driven Functional Discovery

2.1 Training on Data with Full Knowledge

Let the data matrix be

$$M = [v_1, ..., v_N], \tag{1}$$

where $v_i \in \mathbb{R}^m$ is a vector of observations describing the i^{th} data point. Each v_i is allowed to have arbitrarily many missing entries. Define $\text{supp}(v_i) = \{k \in \{1, ..., m\} : v_{i,k} \text{ is observed}\}$.

Due to the missing entries, calculating an affinity between every two points v_i and v_j is not necessarily possible, given that the intersection of the supports of their known values may be small or even disjoint. For this reason, we begin by restricting ourselves to points v_{i_j} that have at most η missing entries. We shall begin by organizing the set of points $\Omega = [v_{i_1}, ..., v_{i_n}]$.

To gain an initial understanding of the geometry of Ω, we consider the cosine affinity matrix A where

$$A_{j,k} = \frac{\langle v_{i_j}, v_{i_k} \rangle}{\|v_{i_j}\| \|v_{i_k}\|}, \tag{2}$$

where the inner product is calculated only on the entries in $\text{supp}(v_{i_j}) \cap \text{supp}(v_{i_i})$. By definition of Ω, this set contains at least $m - 2\eta$ known values.

The cosine affinity matrix serves as a good starting point for learning the geometry of Ω. However, we must develop a way to extend any analysis to the full data M. For this reason, we partition both the data points and the observation sets of Ω. This gives us two advantages: partitioning the data points captures the ways in which different observations may respond to different subsets of Ω, and partitioning the observation sets into similar question groups gives a method for filling in the missing observations in M.

We construct a coupled geometry of Ω using the algorithm developed in [5]. The initial affinity is given by the cosine affinity matrix A, and the iterative procedure is updated using Earth Mover Distance [14].

Remark: Let the final affinity matrix be called $\widetilde{A} : M \times M \to [0, 1]$. Let the eigenpairs of $\widetilde{A}$ be called $\{(\lambda_i, \phi_i)\}$ with $1 = \lambda_0 \geq ... \geq \lambda_N$. Then the organization of M is generated by

$$\Phi^t(x) = [\lambda_1^t \phi_1(x), ..., \lambda_d^t \phi_d(x)], \quad x \in M,$$

where d is the dimension of the underlying manifold.

2.2 Filling in Missing Features

Let $\mathscr{T}_{obs}$ be the hierarchical tree developed on the observations in $\mathbb{R}^m$ from Section 2.1. Let the levels be $\mathscr{X}^1, ..., \mathscr{X}^L$, with the nodes for level l named $\mathscr{X}_1^l, ..., \mathscr{X}_{n(l)}^l$. Let $v_i \notin \Omega$ be a data point with the entry $v_{i,k}$ missing. In order to add v_i into the geometry of Ω, we must estimate the entries in $(\text{supp}(v_i))^c$ to calculate an affinity between v_i and other points.

$\mathscr{T}_{obs}$ gives a tree of correlations between the observations. This allows us to fill in $v_{i,k}$ with similar, known entries. Find the lowest level of the tree (most strongly correlated questions) for which observation $k \in \mathscr{X}_j^l$ and $\exists m \in \mathscr{X}_j^l$ such that $v_{i,m}$ is known. Then the estimate of $v_{i,k}$ satisfies

$$\widetilde{v}_{i,k} = \frac{1}{|\mathscr{X}_j^l|} \sum_{m \in \mathscr{X}_j^l} v_{i,m}. \tag{3}$$

Along with an estimate of $v_{i,k}$, (3) also gives a level of uncertainty for the estimate, as smaller l (i.e. coarser folders) have lower correlation and give larger reconstruction error.

2.3 Expert Driven Function on the Folders

Let $\mathscr{T}_{points}$ be the hierarchical tree developed on the data points in Ω from Section 2.1. Let the levels be $\mathscr{X}^1, ..., \mathscr{X}^L$, with the nodes for level l named $\mathscr{X}_1^l, ..., \mathscr{X}_{n(l)}^l$. As the partitioning becomes finer (i.e. l approaches L), the folders contain more tightly clustered points. This means that the distance from an point to a centroid of a folders becomes smaller as the partitioning becomes finer.

Fix the level l in the tree. The centroids of these folders can be thought of as "types" of data points, or *pseudopoints*. There are two major benefits: there are a small number of

pseudopoints relative to n that span the entire data space, and the pseudopoints are less noisy and more robust to erroneous observations.

These pseudopoints are the key to incorporating expert knowledge and opinion. The pseudopoints are easier and much faster to classify than individual points, as there are a small number and they are less noisy than individual points. Also, the pseudopoints effectively synthesize the aggregate performance of multiple hospitals. The classifications generated by experts can be varied, anything from quality rankings to discrete classes to several descriptive features or "meta-features" of the bins. Specifically, the user assigns a set of classes $\mathscr{C}$ and a classification function $g : \Omega \to \mathscr{C}$ such that

$$\forall x \in \mathscr{X}_j^l, \quad g(x) = y_j \in \mathscr{C}. \tag{4}$$

This function is understood as a rough estimate, since the classification is applied to all $x \in \mathscr{X}_j^l$ even though the class is determined only from the centroid of $\mathscr{X}_j^l$.

This step is non-traditional in unsupervised classification algorithms. Traditionally, the clustering is done agnostic to the final goal of classification. However, this does not allow for a second step correction of the original classification. By allowing a user driven component that quickly examines the cluster centroids, we are able to learn an initial classification map for the points. This provides a rough characterization of which clusters should be collapsed due to similar classification scores, which clusters should be separated further due to drastically different classification scores, and a method of determining class labels for points on the border of multiple clusters.

This function gives a rough metric $\rho : \Omega \times \Omega \to \mathbb{R}^+$ that has dependencies of the form

$$\rho(x, y) = f\left(x - y; g(x), g(y)\right). \quad f : \mathbb{R}^m \times \mathscr{C} \times \mathscr{C} \to \mathbb{R}^+. \tag{5}$$

This metric needs to satisfy two main properties:

1. $\exists \delta_0$ such that $\rho(x, y) < \epsilon_{dist}$ if $\|x - y\| < \delta$ for $\delta < \delta_0$ and some norm $\|\cdot\|$, and
2. $\rho(x, y) > \epsilon_{class}$ if $g(x) \neq g(y)$.

A metric that satisfies these two properties naturally relearns the most important features for preserving clusters while simultaneously incorporating expert knowledge to collapses non-relevant features. We learn this function using neural nets, as described in Section 3.

3 Deep Learning to Form Meta-Features

There are entire classes of functions that approximate the behavior of ρ in (5). For our algorithm, we use neural nets for several reasons. First, the weight vectors on the first layer of

a neural net have clear, physical interpretation, as the weight matrix can be thought of as a non-linear version of the weight vectors from principal component analysis. Second, current literature on neural nets suggest a need for incredibly large datasets to develop meaningful features on unsupervised data. Our algorithm provides a way to turn an unlabeled data set into a semi-supervised algorithm, and incorporates this supervision into the nodes of the neural net. This supervision appears to reduce the number of training points necessary to generate a non-trivial organization. Past literature has used radial basis functions as the non-linear activation for deep learning[4] , as well as for analysis of the structure of deep net representations [12].

3.1 Neural Nets with Back-Propagation of Rankings

For our algorithm, we build a 2 layer stacked autoencoder with a sigmoid activation function, with a regression layer on top. The hidden layers are defined as

$$h^{(l)}(x) = \sigma\left(b^{(l)} + W^{(1)}h^{(l-1)}(x)\right),$$

with $\sigma : \mathbb{R} \to [0, 1]$ being a sigmoid function applied element-wise, and $h^{(0)}(x) = x$. The output function $f(x)$ is logistic regression of these activation units

$$f(x) = \sigma(b^{(3)} + Vh^{(2)}(x)).$$

The reconstruction cost function for training our net is an L^2 reconstruction error

$$C = \frac{1}{n}\sum_{i=1}^{n} \|g(x_i) - f(x_i)\|^2. \tag{6}$$

The overall loss function we minimize, which combines the reconstruction cost with several bounds on the weights, is

$$L = \frac{1}{n}\sum_{i=1}^{n} \|g(x_i) - f(x_i)\|^2 + \mu \sum_{l}\sum_{i,j}\left(W_{i,j}^{(l)}\right)^2$$

Note that we rescale g if it takes values outside $[0, 1]$. We then backpropagate the error by calculating $\frac{\delta L}{\delta w_{i,j}^{(l)}}$ and $\frac{\delta L}{\delta w_{i}^{(l)}}$ and adjusting the weights and bias accordingly. See [13] for a full description of the algorithm.

Definition 3.1. *The deep neural net metric on Ω with respect to an external function f is defined as*

$$\rho_{DNN}(x, y) = \|h^{(1)}(x) - h^{(1)}(y)\|.$$

Lemma 3.2. *A deep neural net with a logistic regression on top generates a metric ρ_{DNN} that satisfies Condition 1 from* (5) *with a Lipschitz constant of $\|W_1\|/4$ with respect to Euclidean distance. The output function f also has a Lipschitz constant of $\|W_1\|\|W_2\|\|V\|/64$ with respect to Euclidean distance.*

Proof. Let $\sigma(x) = \frac{1}{1+e^{-x}}$. Then $\frac{d\sigma}{dx} = \sigma(x)(1-\sigma(x)) \leq \frac{1}{4}$. By the mean value theorem, $|\sigma(a) - \sigma(b)| = |a-b||\frac{d\sigma}{dx}(z)| \leq \frac{|a-b|}{4}$. Then

$$\begin{aligned} \|f(x) - f(y)\|_2 &\leq \|V\|\|h^{(2)}(x) - h^{(2)}(y)\|/4 \\ &\leq \|V\|\|W^{(2)}h^{(1)}(x) - W^{(2)}h^{(1)}(y)\|/16 \\ &\leq \|V\|\|W^{(2)}\|\|h^{(1)}(x) - h^{(1)}(y)\|/16 \\ &\leq \|V\|\|W^{(2)}\|\|W^{(1)}\|\|x-y\|/64. \end{aligned}$$

The same argument applies for ρ_{DNN}.

□

Lemma 3.3. *A two layer neural net with a logistic regression on top creates a function f which satisfies a variant of Condition 2 from* (5)*, namely that*

$$\mathbb{E}_{\neq}\left(\|f(x) - f(y)\|^2\right) \geq \mathbb{E}_{\neq}\left(\|g(x) - g(y)\|^2\right) - 2\left(\frac{\max_{i\in\mathscr{C}} S_i \cdot n}{S}\right)C, \tag{7}$$

where $S = \#\{(x,y) \in \Omega \times \Omega : g(x) \neq g(y)\}$, $S_i = \#\{y \in \Omega : g(y) \neq i\}$, and $\mathbb{E}_{\neq}$ is the expected value over the set S.

Moreover, the deep neural net ρ_{DNN} generated also satisfies

$$\mathbb{E}_{\neq}\left(\|f(x) - f(y)\|^2\right) \geq \left(\frac{1}{\prod_{i=2}^{L}\|W_i\|}\right)\left[\mathbb{E}_{\neq}\left(\|g(x) - g(y)\|^2\right) - 2\left(\frac{\max_{i\in\mathscr{C}} S_i \cdot n}{S}\right)C\right]. \tag{8}$$

Proof. We have

$$\begin{aligned} \|f(x) - f(y)\|^2 &= \|f(x) - g(x) - f(y) + g(y) + g(x) - g(y)\|^2 \\ &\geq \|g(x) - g(y)\|^2 - \left(\|f(x) - g(x)\|^2 + \|f(y) - g(y)\|^2\right). \end{aligned}$$

Unfortunately, because (6) is a global minimization, we cannot say anything meaningful

about the difference for individual points. However, we do have

$$\begin{aligned}\sum_{g(x)\neq g(y)} \|f(x)-f(y)\|^2 &\geq \sum_{g(x)\neq g(y)} \|g(x)-g(y)\|^2 - \sum_{g(x)\neq g(y)} \left(\|f(x)-g(x)\|^2 + \|f(y)-g(y)\|^2\right) \\ &= \sum_{g(x)\neq g(y)} \|g(x)-g(y)\| - 2\sum_{x\in\Omega} \#\{y : g(x)\neq g(y)\}\cdot\|f(x)-g(x)\|^2 \\ &\geq \sum_{g(x)\neq g(y)} \|g(x)-g(y)\| - 2\left(\max_{i\in\mathscr{C}} \#\{y : g(y)\neq i\}\right)\cdot nC,\end{aligned}$$

where $\#\{y : g(y) \neq i\}$ denotes the number of elements in this set. Let $S = \#\{(x,y) \in \Omega\times\Omega : g(x)\neq g(y)\}$ and $S_i = \#\{y\in\Omega : g(y)\neq i\}$. Then

$$\mathbb{E}_{\neq}\left(\|f(x)-f(y)\|^2\right) \geq \mathbb{E}_{\neq}\left(\|g(x)-g(y)\|^2\right) - 2\left(\frac{\max_{i\in\mathscr{C}} S_i\cdot n}{S}\right)C.$$

This means that by minimizing C, we are forcing the separation of points with different initial ranking to be as large as possible. This makes enforces Condition 2 of (5) in the aggregate over all such points.

The scaling of $\frac{1}{\prod_{i=2}^{L}\|W_i\|}$ for ρ_{DNN} is a simple application of Lemma 3.2. □

3.2 Heat Kernel Defined by ρ_{DNN}

The weights generated by the neural net represent "meta-features" formed from the features on Ω. Each hidden node generates important linear and non-linear combinations of the data, and contains much richer information than a single question or average over a few questions.

One downside of traditional deep learning is that neural nets only account for the geometry of the observations. They ignore the geometry and clustering of the data points, opting to treat every point equally by using a simple mean of all points in the cost function (6). This can easily miss outliers and be heavily influenced by large clusters, especially with a small number of training points.

The questionnaire from Section 2.1 organizes both the observations and the data points, though the meta-features from the questionnaire are simply averages of similar features as in (3). However, the expert ranking assigned to each bin reflects a rough initial geometry onto the points to be learned by the neural net.

The back propagation of the expert driven function is essential to building significant weight vectors in this regime of small datasets. When the ratio of the number of data points to the dimension of the features is relatively small, there are not enough training points to learn

the connections between all the features without considering the points themselves. The back propagation of the classification function generated from the questionnaire is a way to enforce the initial geometry of the data on the weight vectors. The hospital ranking example in Section 4 demonstrates the need for back propagation, and Figure 7 demonstrates the fact that the features from a simple SDAE are not sufficient for separating data points.

For our algorithm, the SDAE is trained on Ω, the subset of points with "full" collection of features. This is to avoid training on reconstructed features which are subject to reconstruction error from Section 2.2.

Another problem with neural nets is that they can be highly unstable under parameter selection (or even random initialization). Two identical iterations can lead to completely different weights set. Along with that, back propagation can force points into isolated corners of the cube in $[0,1]^k$.

For this reason, we rerun the neural net K times with varied random seeds, number of hidden layers, sparsity parameters, and dropout percentages. After K iterations, we build the new set of features on points as $\Omega^*(x) = [h_1^{(1)}(x), ..., h_K^{(1)}(x)]$. This defines an adjacency matrix on A with affinity defined between two points as

$$A(x,y) = e^{-\|\Omega^*(x)-\Omega^*(y)\|^2/\epsilon}. \tag{9}$$

Along with that, the final ranking function on M comes from $f(x) = \frac{1}{K}\sum_{i=1}^K f_i(x)$. Note that $\|\Omega^*(x) - \Omega^*(y)\|^2 = \sum_{i=1}^K \rho_{DNN,i}(x,y)^2$.

The expert driven heat kernel defined in (9) generates an embedding $\Phi : \Omega \to \mathbb{R}^d$ via the eigenvectors $\Phi^t(x) = [\lambda_1^t\phi_1(x), ..., \lambda_1^t\phi_d(x)]$.

For each neural net h_i, we keep the number of hidden layers small relative to the dimension of the data. This keeps the net from overfitting the data to the initial organization function g.

3.3 Standardizing Distances to Build an Intrinsic Embedding

While this generates a global embedding based off local geometry, it does not necessarily generate a homogenous space. In other words, $\|\Phi^t(x) - \Phi^t(y)\| = \|\Phi^t(x') - \Phi^t(y')\|$ does not necessarily guarantee that x and y differ by same amount as x' and y'. This is because $\Omega^*(x)$ and $\Omega^*(y)$ may differ in a large number of deep net features, whereas $\Omega^*(x')$ and $\Omega^*(y')$ may only differ in one or two features (though those features may be incredibly important for differentiation).

For this reason, we must consider a local z-score of the regions of the data. For each point $\Phi^t(x)$, there exists a mean and covariance matrix within a neighborhood U_x about x such

that

$$\begin{aligned}\mu_x &= \frac{1}{|U_x|}\sum_{z\in U_x}\Phi^t(z),\\ \Sigma_x &= \frac{1}{|U_x|}\sum_{z\in U_x}\left(\Phi^t(z)-\mu_x\right)^\intercal\left(\Phi^t(z)-\mu_x\right).\end{aligned}$$

This generates a new whitened distance metric

$$d_t(x,y)=\frac{1}{2}\left[(\Phi^t(x)-\mu_x)-(\Phi^t(y)-\mu_y)\right]^\intercal\left(\Sigma_x^\dagger+\Sigma_y^\dagger\right)\left[(\Phi^t(x)-\mu_x)-(\Phi^t(y)-\mu_y)\right], \quad (10)$$

where $\Sigma^\dagger$ is the Penrose-Moore pseudoinverse of the covariance matrix.

One can generate a final, locally standardized representation of the data via the diffusion kernel

$$W(x,y)=e^{-d_t(x,y)/\sigma},$$

and the low frequency eigenvalues/eigenvectors of W, which we call $\{(s_i,\psi_i)\}$. The final representation is denoted

$$\Phi^t_{std}(x)=[s_i^t\psi_1(x),...,s_i^t\psi_d(x)]. \quad (11)$$

3.4 Extension to New Points

Once meta-features have been built, extending the embedding to the rest of M can be done via an asymmetric affinity kernel, as described in [9]. For each $x\notin\Omega$, one can calculate $\Omega^*(x)=[h_i^{(1)}(x)]_{i=1}^K$ and $f_i(x)$ with the weights generated in Section 3.2. This defines an affinity between x and $y\in\Omega$ via

$$a(x,y)=e^{-\|\Omega^*(x)-\Omega^*(y)\|^2/\epsilon}.$$

The affinity matrix is $a\in\mathbb{R}^{N\times n}$. [9] then shows that the extension of the eigenvectors ϕ_i of A from (9) to the rest of M is given by

$$\widetilde{\phi}_i=\frac{1}{\lambda_i^{1/2}}\widetilde{A}\phi_i,$$

where $\widetilde{A}$ is the row stochastic normalization of a.

Similarly, once the embedding Φ is extended to all points in M, we can extend the final, standardized representation to the rest of M as well using the affinity matrix

$$b(x,y)=exp\{-\frac{1}{2}\left[(\Phi^t(x)-\mu_y)-(\Phi^t(y)-\mu_y)\right]^\intercal\Sigma_y^\dagger\left[(\Phi^t(x)-\mu_y)-(\Phi^t(y)-\mu_y)\right]/\sigma\},$$

for $x \in M$ and $y \in \Omega$. Once again, the eigenvectors ψ_i in (11) can be extended similarly via

$$\widetilde{\psi}_i = \frac{1}{s_i^{1/2}} \widetilde{B} \psi_i,$$

where $\widetilde{B}$ is the row stochastic normalized version of b.

In this way, Φ^t_{std} can be defined on all points M after analysis on the reference set Ω.

3.5 Different Approach to Deep Learning

Our algorithm, as we will discuss in detail in Section 3, uses the meta-features from stacked neural nets in a way not commonly considered in literature. Most algorithms use back propagation of a function to fine-tune weights matrices and improve the accuracy of the one dimensional classification function. However, in our algorithm, the purpose of the back propagation is not to improve classification accuracy, but instead to organize the data in such a way that is smooth relative to the classification function. In fact, we are most interested in the level sets of the classification function and understanding the organization of these level sets.

At the same time, we are not building an auto-encoder that pools redundant and correlated features in an attempt to build an accurate, lossy compression (or expansion) of the data. Due to the high level of disagreement among the features, non-trivial features generated from an auto-encoder are effectively noise, as we see in Section 4.2. This is the motivation behind propagating an external notion of "quality".

4 Expert Driven Embeddings for Hospital Rankings

4.1 Hospital Quality Ranking

For preprocessing, every feature is mean centered and standard deviation normalized. Also, some of the features are posed in such a way that low scores are good, and others posed such that low scores are bad. For this reason, we "de-polarize" the features using principal component analysis. Let $M = [v_1, ..., v_N]$ be the data matrix. Then let U be the largest eigenvector of the covariance matrix

$$\begin{bmatrix} M \\ -M \end{bmatrix} \times \begin{bmatrix} M^* & -M^* \end{bmatrix},$$

and take the half of the features i for which $U_i > 0$. This makes scores above the mean "good" regardless of whether the feature is posed as a positive or negative question.

We begin by building a questionnaire on the hospitals. Our analysis focuses on the 2014 CMS measures.We put a 2× weight on mortality features and 1.5× weight on readmission features due to importance, because the outcome measures describe the tangible results of a hospitalization are particularly important for patients. The questionnaire learns the relationships between the hospitals, as well as the relationships between the different features. In doing this, we are able to build a partition tree of the hospitals, in which hospitals in the same node of the tree are more similar than hospitals in different nodes on the same level of the tree. As a side note, the weighting on mortality features only guarantees that the intra-bin variance of the mortality features is fairly low.

For the ranking in this example, we use the 5^{th} layer of a dyadic questionnaire tree, which gives 32 bins and pseudohospitals. Experts on CMS quality measures rank these pseudohospitals on multiple criteria and assign a quality score between 1 to 10 to each pseudohospitals. We use an average of 50 nodes per layer of the neural net. Also, we average the results of the neural nets over 100 trials. We shall refer to the final averaged ranking as the deep neural net ranking (DNN ranking) to avoid confusion.

4.2 Two Step Embedding of Hospitals

Figure 4 shows the embedding of the subset of hospitals used in training the neural net. It is colored by the quality function assigned to those hospitals. The various prongs of the embedding are explained in Section 4.3.4. Figure 5 shows the embedding of the full set of hospitals. This was generated via the algorithm in Section 3.4.

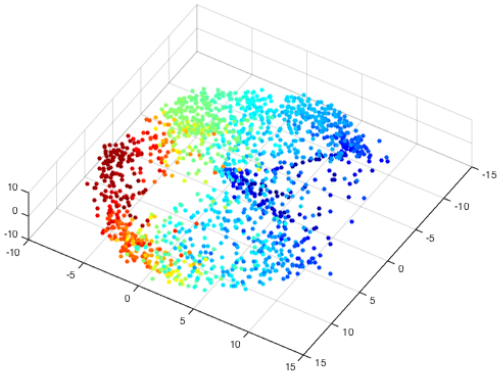

Figure 4: Embedding of hospitals with at most τ missing entries. Red corresponds to top quality, blue to bottom quality.

Finally, while the goal of the hospital organization is to determine neighborhoods of similar

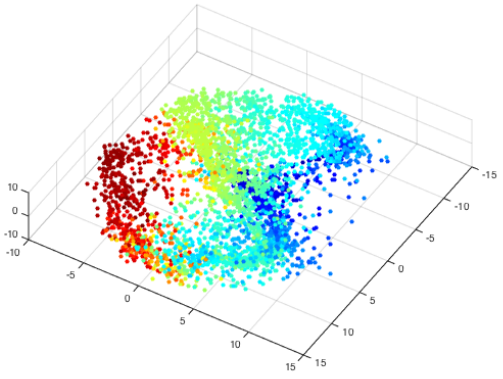

Figure 5: Embedding of full set of hospitals. Red corresponds to top quality, blue to bottom quality.

hospitals, it is also necessary for all hospitals in a shared neighborhood to share a common quality rating. Figure 6 plots the quality function assigned to the hospitals against the weighted average quality function of its neighborhood, where the weights come from the normalized affinities between the given hospitals and its neighbors. The strong collinearity demonstrates that the assigned quality function is consistent within neighborhoods of similar hospitals.

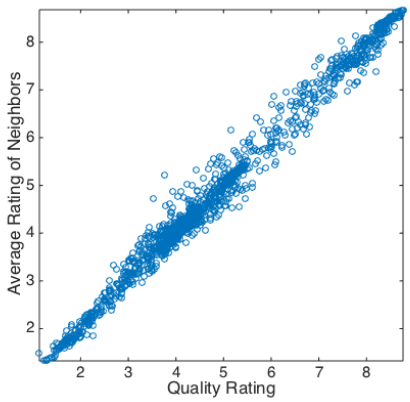

Figure 6: Quality function assigned for hospital versus average quality across the neighborhood in Figure 4.

To demonstrate that the expert input back propagation is necessary for a viable ranking and affinity, we include Figure 7. Here, we build the same diffusion map embedding, but

on the features of the autoencoder before back propagation of the expert input function. Due to the small number of data points relative to the number of features, an untuned autoencoder fails to form relevant meta-features for the hospitals.

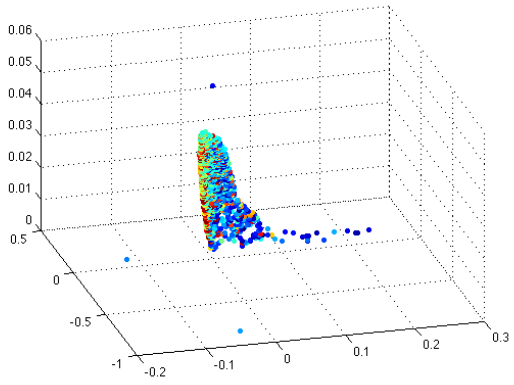

Figure 7: Embedding of Ω without back propagation of expert driven hospitals rankings. Coloring comes from quality rating function. Notice that without back propagation the embedding is effectively noise compared to hospital quality.

4.3 Internal Validation

The problem with validating this composite rating of hospital quality is the lack of external ground truth. For this reason, we must rely on internal validation mechanisms to demonstrate consistency. There are two qualities necessary for a ranking mechanism: the affinity between hospitals cannot disagree too drastically with the original features, and similar hospitals must be of similar quality.

For the rest of the section, we refer to several simple meta-features for comparison and validation. Row sum ranking refers to a simple ranking function

$$f_{row-sum}(x) = \sum_i x_i,$$

where $\{x_i\}$ are the normalized and depolarized features for a given hospital x. Average process, survey, and outcome features are the same as the row sum score, but restricted only to features in the given category. NNLS weighted ranking refers to a function

$$f_{NNLS}(x) = \sum_i w_i x_i,$$

where the weights are calculated using non-negative least squares across the normalized and depolarized features, with the DNN ranking as the dependent variable. NNLS weighted process, survey, and outcome features are the same as the NNLS weighted score, but restricted only to features in the given category. NNLS was used for calculating the weights to avoid overfitting the DNN ranking by using negative weights.

Also, our focus (unless otherwise stated) is on the subset of hospitals Ω that have more than 90% features reported. This is because a number of our validation steps use the row sum ranking and NNLS weighted ranking, which are more accurate when calculated over non-missing entries.

4.3.1 DNN Satisfies Conditions for ρ

Figure 8 verifies that the back propagation neural net satisfies Condition 1 of ρ from (5). The plot shows the ranking of each hospital plotted against the average of its ten nearest neighbors under Euclidean distance between hospital profiles. The fact that the average strongly correlates with the original quality rating shows that the embeddings of the hospitals remain close if they are close under Euclidean distance.

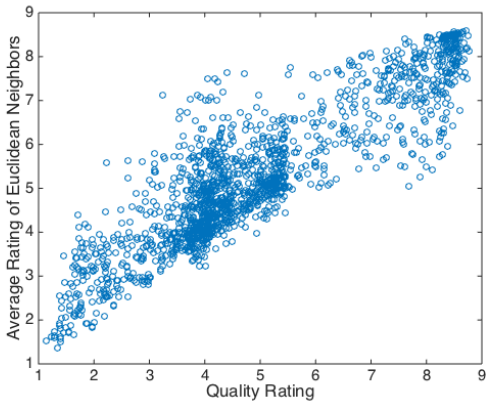

Figure 8: Quality function for hospital versus average quality function across closest Euclidean points. Demonstrates first condition necessary for ρ from (5).

Figure 9 shows that the back propagation neural net satisfies Condition 2 of ρ from (5). The histogram on the left shows the DNN affinity between hospitals with different initial quality ratings (i.e. when $g(x) \neq g(y)$). To satisfy Condition 2, $g(x) \neq g(y) \implies \rho(x, y) > \epsilon_{class}$ (i.e. $A(x, y) < 1 - \epsilon$). Also, for the histogram on the right we define

$$P_{\neq}(t) = \Pr\left(A(x, y) > t : g(x) \neq g(y)\right), \quad P_{=}(t) = \Pr\left(A(x, y) > t : g(x) = g(y)\right).$$

The histogram on the right shows $\frac{P_{\neq}(t)}{P_{=}(t)}$. The smaller the ratio as t approaches 1, the stronger the influence of the initial ranking on the final affinity.

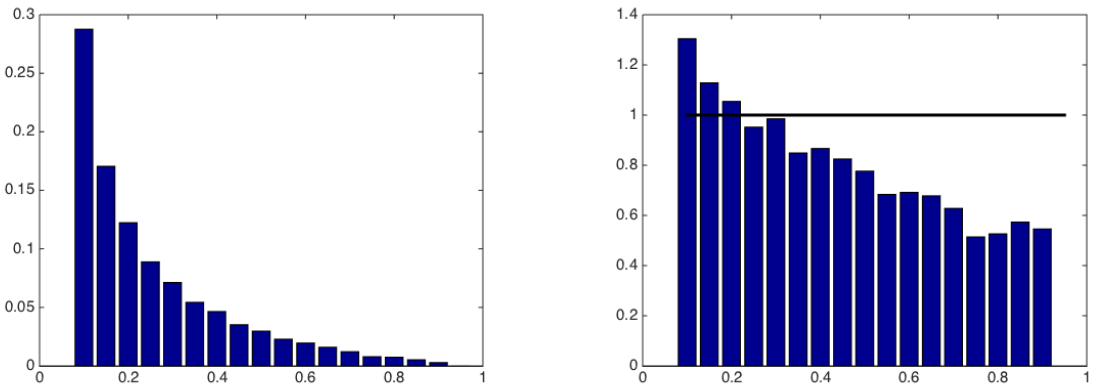

Figure 9: (left) Histogram of DNN affinity between two points with different initial quality ratings. (right) Normalized histogram of DNN affinity between two points with same initial quality rating. If initial ranking was unimportant, bar graph would be concentrated around 1. Both images demonstrate second condition necessary for ρ from (5). Note that, for scaling purposes, all values less than .1 have been removed from counting.

Moreover, we can compare the contraction guaranteed by (8). For the hospital ratings,

$$\mathbb{E}_{\neq}\left(\|f(x)-f(y)\|^2\right)=2.44,\ \mathbb{E}_{\neq}\left(\|g(x)-g(y)\|^2\right)=3.36,\quad \frac{\max_{i\in\mathscr{C}} S_i\cdot n}{S}=1.22,\ C=0.9959,$$

which makes the right hand side of (8) equal 2.15.

Figure 10 indicates that the DNN metric from (5) gives a better notion of small neighborhoods than a simple Euclidean metric. Each metric defines a transition probability $P(x,y)$. For each point x, the plot finds

$$\begin{aligned} \underset{I\subset\Omega}{\text{minimize}} \quad & \#I \\ \text{subject to} \quad & \sum_{y\in I} P(x,y)\geq\frac{1}{2}. \end{aligned} \tag{12}$$

It is important for (12) to be small, as that implies the metric generates tightly clustered neighborhoods. As one can see from Figure 10, the DNN metric creates much more tightly clustered neighborhoods than a Euclidean metric. Figure 11 gives summary statistics of these neighborhoods for varying diffusion scales.

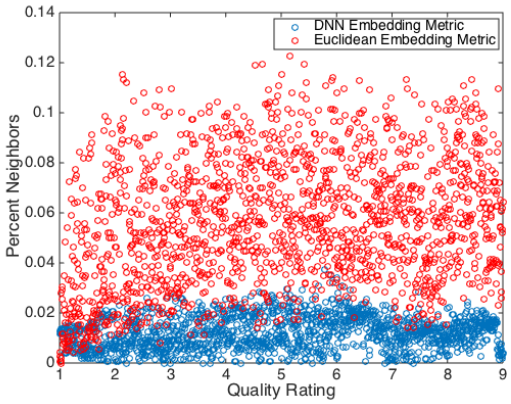

Figure 10: Percent of points necessary to sum half the total transition probability. Metrics are generated in same fashion: $K(x,y) = e^{-\|x-y\|^2/\sigma^2}$. Here $\sigma = \frac{1}{N}\sum_x \|x - y_x\|$, where y_x is the 10^{th} closest neighbor of x.

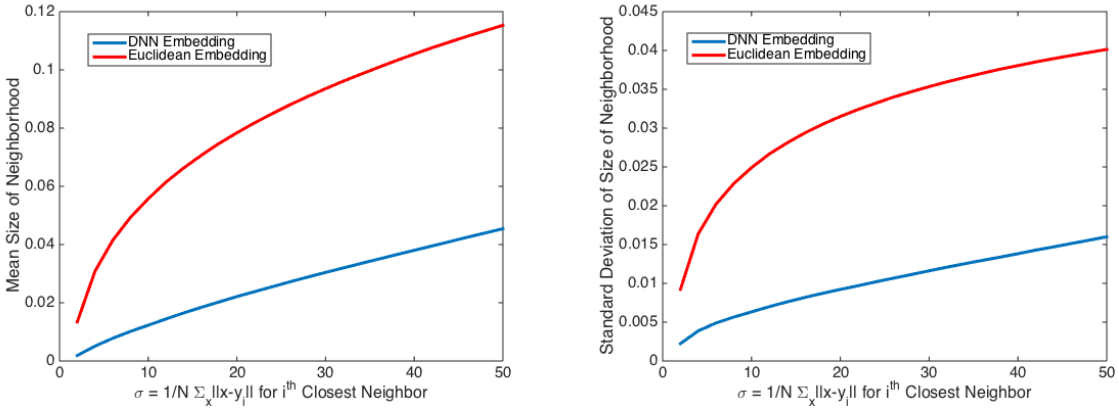

Figure 11: Metrics are generated in same fashion: $K(x,y) = e^{-\|x-y\|^2/\sigma_i^2}$. Here $\sigma_i = \frac{1}{N}\sum_x \|x - y_x\|$, where y_x is the i^{th} closest neighbor of x. For both DNN embedding and Euclidean embedding metrics, the plots are (left) the average percentage of neighbors needed to contain half the transition probability from a given point to its neighborhood, and (right) the standard deviation of the number of neighbors needed to contain half the transition probability from a given point to its neighborhood.

Another positive characteristic of our DNN embedding is the reduced local dimension of the data. Figure 12 plots the eigenvalues $\{S_i^t\}_{i=0}^n$ of the diffusion kernels A_t, where the Markov chain A_t describes either comes from the Euclidean metric or the DNN metric. These eigenvalues give us information about the intrinsic dimension of the data, as small

eigenvalues do not contribute to the overall diffusion. To compare the eigenvalues across different diffusion kernels, we normalize the eigenvalues by setting $t = \frac{1}{1-S_1^1}$, as this is the average time it takes to diffuse across the system. Cutting off the eigenvalues to determine the dimension is fairly arbitrary without a distinct drop off, but an accepted heuristic is to set the cutoff at

$$\dim(A_t) = \max\{d \in \mathbb{N} : S_d > 0.01\}.$$

With this definition of dimension, $\dim(A_t) = 7$ for the DNN metric embedding, whereas $\dim(A_t) = 14$ for the Euclidean metric embedding.

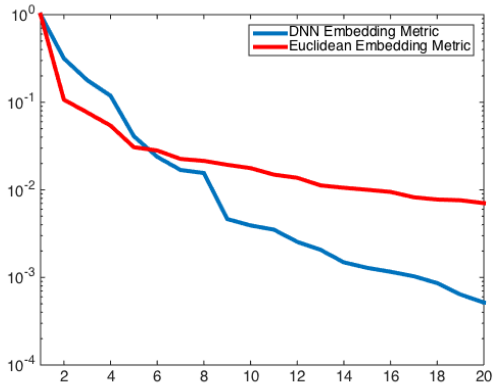

Figure 12: Eigenvalues $\{S_i^t\}_{i=0}^n$ of the embedding of the heat kernel A_t generated from the Euclidean metric (blue) and DNN metric (red). Each eigenvalue set is normalized by setting $t = \frac{1}{1-S_1^1}$.

4.3.2 Examples of "Best" and "Worst" Hospitals

Row sum ranking is a poor metric for differentiation between most hospitals, but it is able to distinguish the extremal hospitals. In other words, hospitals with almost all features considerably above the mean would have obviously been assigned a 10 within any ranking system, and hospitals with almost all features below the mean would have obviously been assigned a 1 by any ranking system.

Let us take the 25 best and worst hospitals under the row sum ranking, and consider how well these correspond to the final DNN rankings from Section 4.2. The average quality function value across the top 25 row sum rankings is 8.03, and the average quality function value across the bottom 25 row sum rankings is 1.75.

While these averages are indicative of the agreement between the DNN ranking and simple understandings of hospital quality, they are not perfect. For example, the top 25 row sum hospitals have an average DNN ranking of 8.03 because one of these hospitals has a DNN ranking of 4.84. This hospital is an interesting profile. Qualitatively, its process features and survey features are above average, but their outcomes are significantly worse than the mean. Even though only 16 of its 81 features are below the mean, the features for mortality from heart failure and pneumonia, and several measures of hospital associated infection, are all more than 1 standard deviation below the mean. As all five of those features are heavily weighted by the algorithm and deemed crucial by experts, the hospital is assigned a low ranking despite having a strong row sum ranking.

Figure 13 shows the embeddings of hospitals without the neural net step. In other words, we simply weight the mortality and readmission features and calculate an affinity based off those feature vectors. There are two important details. First, the embedding is very unstructured, implying a lack of structure within the naive affinity matrix. Second, while the cloud doesn't necessarily reflect structure, it does give a simple understanding of local neighborhoods.

Figure 13 also shows that our new notion of rankings reflects a naive understanding of hospital quality. Namely, local Euclidean neighborhoods maintain the same ranking. Also, the 10 best and worst hospitals under the row sum ranking are circled. As expected, they exist on the two extremes of the embedding cloud, and our rankings agree with their notion of quality.

4.3.3 Dependence on Initial Rankings

Another important feature of a ranking algorithm is its robustness across multiple runs. To demonstrate stability of the neural net section of the algorithm, we run multiple experiments with random parameters to test the stability of the quality function. Figure 14 shows the quality rating from one run of the algorithm (ie. the average ranking after 100 iterations of the neural net) against the quality rating from a separate run of the algorithm. In both runs, $1,000$ hospitals from Ω are randomly chosen for training the model and then tested on the other 614 hospitals in Ω. Figure 14 shows the rankings for all $1,614$ hospitals across both runs. Given the strong similarity in both quality rating and overall organization, it is clear that the average over 100 iterations of a neural net is sufficient to decide the quality function.

It is also important to examine the dependence our initial binning and ranking has on the final ranking of the hospitals. Clearly the initial binning is only meant to give approximate ranks, so a strong dependence on these rankings would be problematic. Table 1 shows the confusion matrix between the initial hospital rankings and the final rankings assigned from

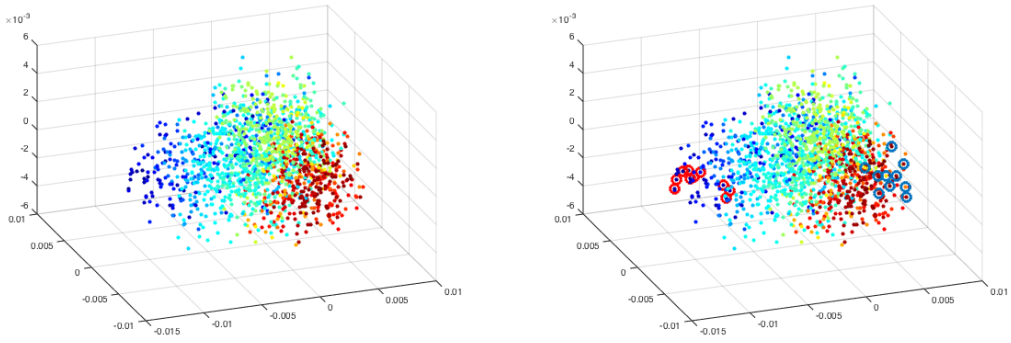

Figure 13: Left: Embedding of hospitals using Euclidean distance on profiles, with 2× weighting on morality features and 1.5× weighting on readmission features. Coloring is value of the quality function described in Section 4.2, with top ratings being red and bottom ratings being blue. Right: Same embedding, with extremal points circled. Blue circles on 10 hospitals with largest number of features above the mean, and red circles on 10 hospitals with fewest number of features above the mean. This is a rough characterization of "best" and "worst" hospitals.

	Final Rank			
First Rank	8	11	0	0
	15	730	129	0
	0	97	330	219
	0	0	27	48

Table 1: Confusion matrix between initial ranking on hospital bins and final ranking from neural net. For simplicity, rankings have been rounded into quartiles for purposes of confusion matrix.

the neural net. The purpose of the neural net second step is to reclassify hospitals that are binned incorrectly due to spurious correlations.

Let us examine a few of the hospitals that made the biggest jump. One of the hospitals increased from a rating of 3 to a DNN rating of 6.55. Several of its process features are well below average with a couple 2 standard deviation below average, and its survey features are average to slightly below average. For this reason, it was immediately classified as a poor hospital. However, its readmission features across the board are above average, with readmission due to heart failure 1.5 standard deviations above average. Also, mortality due to heart failure is a standard deviation above average, and all features of hospital associated infection are above the mean as well. In the initial ranking of the pseudohospitals, the fact that there are 29 survey features as compared to 5 readmission features brought the

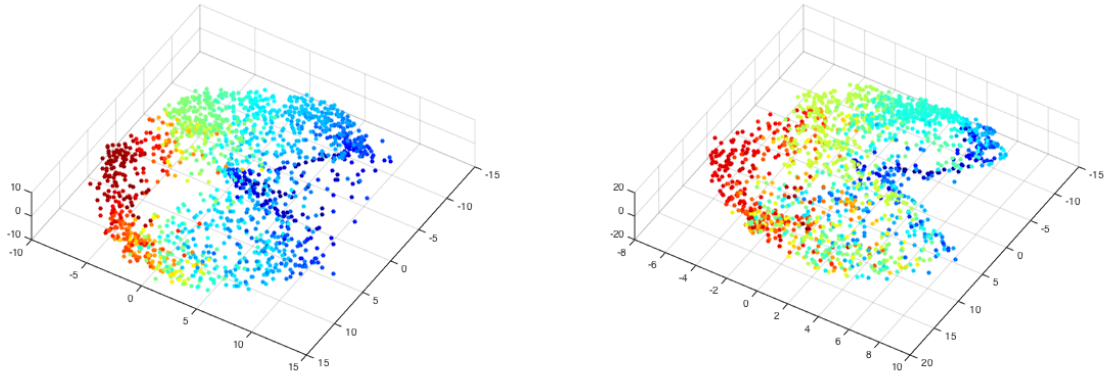

Figure 14: (left) Original embedding colored by quality function. Same image as Figure 4. (right) New embedding of 100 nets trained independently from first run. Embedding is finally rotated to match orientation of the original via [6]. Quality rating coloring the second embedding is generated independently of left embedding.

weighting down, even when giving extra weight to the readmission features. After the neural net ranking, the ranking is improved to 6.55.

Another hospital dropped from a rating of 7 to a DNN rating of 3.4. Its process feature are all above average, and its readmission for heart failure is 1.5 standard deviations above the mean. Because readmission is weighted initially and there are a large number of process measures, the hospital was initially binned with better hospitals. However, its features for hospital associated infections, as well as mortality due to pneumonia and heart failure are a standard deviation below the mean, and its average survey score is a standard deviation below the mean as well. For these reasons, the DNN ranking dropped significantly.

4.3.4 Affinity Dependence on Individual Features

As another validation mechanism, we consider the smoothness of each feature against the metric ρ. For each feature f_i, we estimate the Lipschitz constant L_i such that

$$|f_i(x) - f_i(y)| \leq L_i \|x - y\|_{DNN},$$

where $\|\cdot\|_{DNN}$ is simply the Euclidean distance in the diffusion embedding space of the DNN affinity matrix A. Once f is normalized by range(f), L_i gives a measure of the smoothness of f_i under our new metric.

Lipschitz Constant Under ρ

Feature	Lipschitz Constant
Quality Rating Function	0.124
READM 30 HOSP WIDE	0.303
MORT 30 HF	0.346
READM 30 HF	0.350
MORT 30 AMI	0.371
MORT 30 PN	0.384
READM 30 PN	0.398
READM 30 AMI	0.430
READM 30 HIP KNEE	0.440
H HSP RATING 0 6	0.582
H RECMND DN	0.595
H COMP 1 SN	0.618
H HSP RATING 9 10	0.620
H COMP 3 SN	0.688
H RECMND DY	0.691
H COMP 1 A	0.712
H QUIET HSP SN	0.735
VTE 5	0.740
PSI 90 SAFETY	0.758
H COMP 2 SN	0.758
H COMP 4 SN	0.761
COMP HIP KNEE	0.762
STK 8	0.763
ED 1b	0.766
OP 20	0.789
ED 2b	0.795
PSI 4 SURG COMP	0.807
VTE 3	0.807
H RECMND PY	0.811
H HSP RATING 7 8	0.831
OP 21	0.832
STK 1	0.834
HAI 5	0.838
H CLEAN HSP SN	0.839
H COMP 5 SN	0.844
HAI 2	0.846
HAI 3	0.868
H COMP 6 Y	0.870
H COMP 6 N	0.870
HAI 1	0.875
STK 6	0.889
H COMP 4 A	0.898
PC 01	0.904
VTE 2	0.906

Feature	Lipschitz Constant
H COMP 1 U	0.908
OP 11	0.913
OP 9	0.913
H COMP 3 A	0.918
H COMP 2 A	0.931
OP 13	0.934
OP 18b	0.935
STK 10	0.942
H QUIET HSP A	0.946
OP 10	0.978
H COMP 5 A	0.996
H CLEAN HSP A	1.023
HF 3	1.077
AMI 2	1.081
HAI 6	1.088
SCIP INF 9	1.102
VTE 4	1.106
AMI 10	1.133
OP 6	1.147
H COMP 2 U	1.186
SCIP CAR	1.192
VTE 1	1.221
H CLEAN HSP U	1.225
OP 22	1.227
H QUIET HSP U	1.258
SPP	1.264
IMM 1A	1.272
IMM 2	1.284
STK 2	1.288
OP 7	1.291
H COMP 4 U	1.325
H COMP 3 U	1.365
HF 1	1.430
STK 5	1.443
PN 3b	1.545
H COMP 5 U	1.566
SCIP INF 2	1.994
SCIP INF 3	2.001
PN 6	2.066
SCIP INF 1	2.131
SCIP VTE	2.151
SCIP INF 10	3.333
HF 2	5.111

From the Lipschitz constant, we can conclude that the most influential features are the mortality and readmission scores, followed closely by survey scores and post surgical

infection scores. While the process features have some smoothness, the Lipschitz constants are significantly larger than outcome and some survey scores.

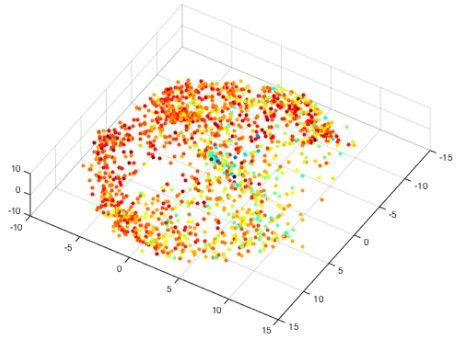

Figure 15: Hospital Diffusion embedding colored by average across 34 process features.

Almost all the hospitals in Figure 15 have an average process score within a half standard deviation of the mean, which is well within acceptable levels. For this reason, these features are not as strong of features as some of the other features. However, the few hospitals that are well below the mean have middle to poor quality ratings.

Figures 16 and 17 show the embedding of the hospitals colored by their average survey and average outcome features. These are more strongly correlated with the overall quality function.

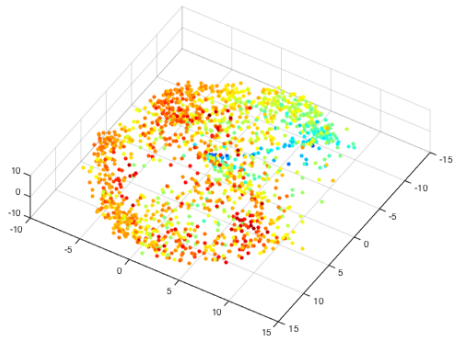

Figure 16: Hospital Diffusion embedding colored by average across 29 survey features.

Note that the average across all features from a category is not the ideal description of that category. For a given hospital, there can be significant variability within a category

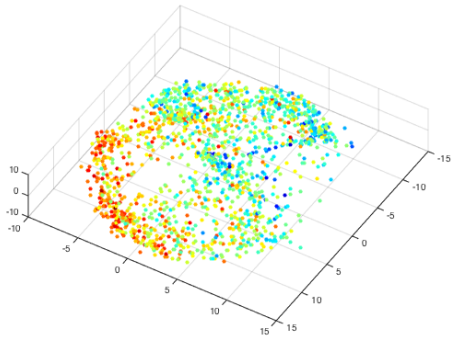

Figure 17: Hospital Diffusion embedding colored by average across 15 outcome features.

which would not be captured by the mean. The category average is only meant to demonstrate a general trend. Figure 18 shows a weighted average of the readmission and mortality features, with the weights determined by a least squares fit with the DNN quality function as a dependent variable.

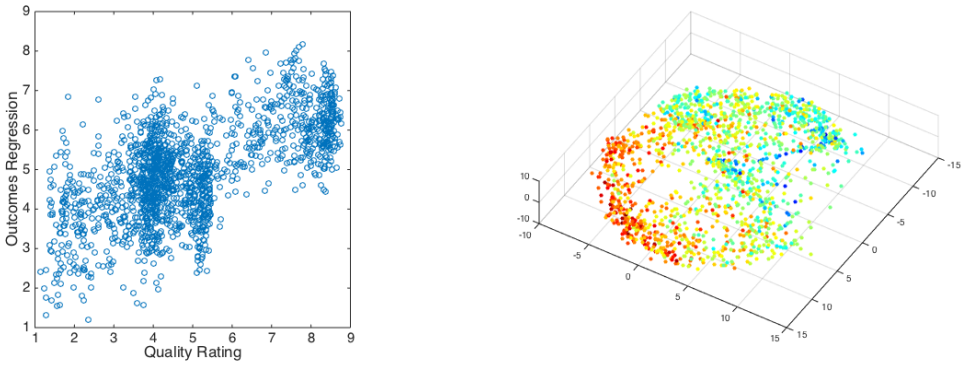

Figure 18: (left) Scatter plots of quality function vs least squares solution with mortality and readmission as independent variables and quality function as dependent variable. (right) Hospital Diffusion embedding colored by same least squares solution.

We can also use these embeddings, along with Figure 19, to show why the embedding has the two prongs on the left side of the image among the well ranked hospitals. Notice from Figure 19 that readmission is very strong along the lower prong, while mortality scores are very strong along the upper prong. Moreover, there is some level of disjointness between the two scores [10].

This begs the question of why the strong readmission cluster is ranked higher than the

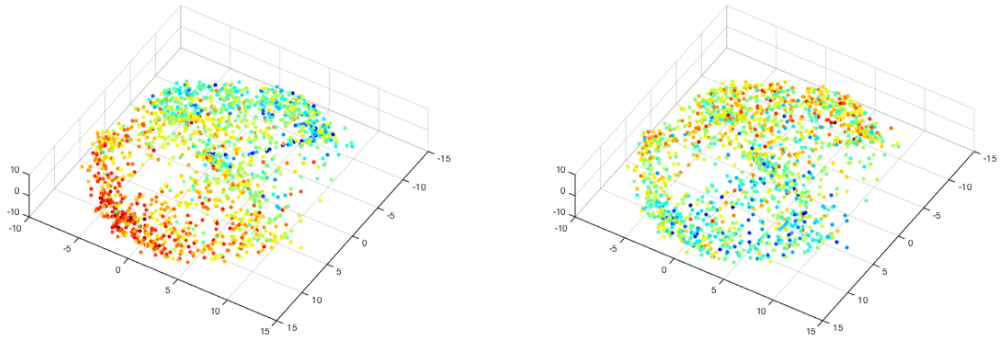

Figure 19: (left) Hospital Diffusion embedding colored by hospital wide readmission (right) Hospital Diffusion embedding colored by mortality from heart failure.

strong mortality cluster. That can be explained in Figure 16. The survey features are much stronger along the lower cluster with high readmission than they are in the mortality cluster. This explains the cluster of hospitals receiving top quality ratings despite having average to slightly above average mortality scores. The same can be said, to a lesser degree, for the process scores in Figure 15. So the differentiation in rankings is derived from the fact that, at their highest levels, more of the features agree with readmission features than they do with mortality features.

The left plot in Figure 20 represents each hospital with three features: the mean score across process features, the mean score across survey features, and the mean score across outcome features. The plot is colored by the quality function, and it is clear that the rankings reflect the trend of these three average features. The right plot in Figure 20 represents each hospital with the NNLS process, survey, and outcome scores. Table 2 list those features that have non-trivial weights with p-value $< .05$.

5 Conclusion

We introduced an algorithm for generating new metrics and diffusion embeddings based off of expert ranking. Our algorithm incorporates both data point geometry via hierarchical diffusion geometry and non-linear meta-features via stacked neural nets. The resulting embedding and rankings represent a propagation of the expert rankings to all data points, and the resulting metric generated by the stacked neural net gives a Lipschitz representation with respect to Euclidean distance that learns important and irrelevant features according to the expert opinions and in automated fashion.

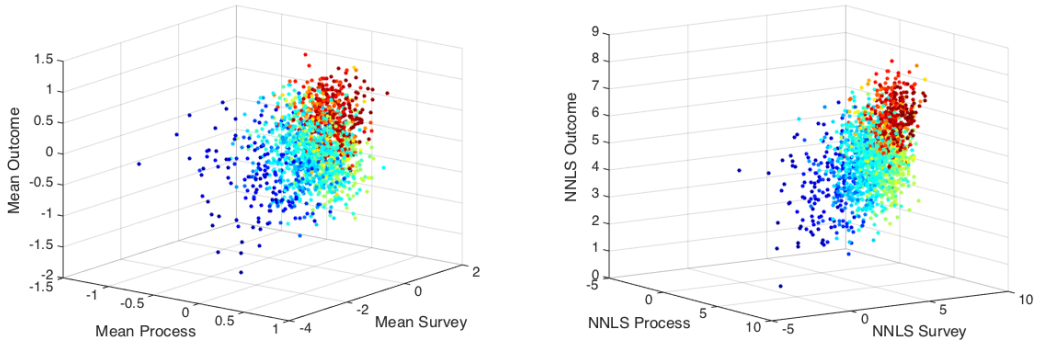

Figure 20: (left) Plot of mean scores across process, survey, and outcome measures, colored by the quality function. (right) Plot of process, survey, and outcome measures with weights determined by non negative least squares with the quality function as dependent variable, colored by quality rating.

Although the ranking algorithm seems tied to the process of expert rankings of hospitals, the underlying idea of generating metrics in the form of (5). and propagating first pass rankings to the rest of the data, is quite general. For example, this method could be used to propagate sparsely labeled data to the rest of the dataset, with expert bin rankings replaced by the mode of the labelled data in the bin.

This method also touches on the importance of incorporating data point organization into the neural net framework when dealing with smaller data sets and noisy features. Without the expert driven function to roughly organize the data points, the stacked autoencoder fails to determine any relevant features for separation, as shown in Figure 7.

We will examine the implications of our hospital ratings on health policy, as well as discuss the various types of hospitals in our embedding, in a future paper. Future work will also examine further examination of the influence of data point organization on neural nets and the generation of meta-features. Also, it would be interesting to examine other applications of propagation of qualitative rankings and measures.

Acknowledgments

The authors would like Arjun K. Venkatesh MD, MBA, MHS and Elizabeth E. Drye MD, SM for helping to develop the initial rankings of the pseudohospitals, Uri Shaham for use of his deep learning code, and Ali Haddad for providing the base code for the questionnaire. Alexander Cloninger is supported by NSF Award No. DMS-1402254.

Features	Coefficient Magnitude
HAI 3	0.029
HAI 6	0.010
READM 30 AMI	0.148
READM 30 HF	0.139
READM 30 HIP KNEE	0.148
READM 30 HOSP WIDE	0.122
READM 30 PN	0.074
PSI 4 SURG COMP	0.018
PSI 90 SAFETY	0.003
MORT 30 HF	0.116
MORT 30 PN	0.121
MORT 30 AMI	0.072
SPP	0.053
OP 10	0.045
OP 11	0.080
OP 13	0.003
AMI 10	0.069
HF 1	0.058
HF 3	0.136
STK 1	0.136
STK 5	0.040
STK 6	0.156
STK 8	0.059
STK 10	0.083
VTE 2	0.183
VTE 3	0.126
VTE 4	0.002
PN 6	0.112
SCIP CAR	0.074
IMM 2	0.239
OP 6	0.077
OP 7	0.009
OP 21	0.028
PC 01	0.108
H CLEAN HSP SN	0.171
H COMP 1 SN	0.141
H COMP 2 SN	0.082
H COMP 5 SN	0.119
H COMP 6 Y	0.156
H HSP RATING 7 8	0.185
H RECMND DN	0.176
H RECMND PY	0.123

Table 2: Significant features from Non Negative Least Squares with Quality Function as the Dependent Variable

References

[1] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *IEEE Transactions on Neural Computation*, 2003.

[2] Yoshua Bengio. Deep learning of representations: Looking forward. *Statistical Language and Speech Processing*, 2013.

[3] E.H. Bradley, J. Herrin, B. Elbel, and et al. Hospital quality for acute myocardial infarction: Correlation among process measures and relationship with short-term mortality. *JAMA*, 2006.

[4] Y. Cho and LK Saul. Kernel methods for deep learning. *Advances in neural information processing systems*, 2009.

[5] Ronald R. Coifman and Matan Gavish. Harmonic analysis of digital data bases. *Wavelets and Multiscale analysis*, 2011.

[6] Ronald R Coifman and Matthew J Hirn. Diffusion maps for changing data. *Applied and Computational Harmonic Analysis*, 2013.

[7] Ronald R Coifman and Stéphane Lafon. Diffusion maps. *Applied and computational harmonic analysis*, 21(1):5–30, 2006.

[8] Center for Medicare and Medicaid Services (CMS). Hospital compare. *CMS website: www.medicare.gov/hospitalcompare/*, Access Date: June 15, 2015.

[9] Ali Haddad, Dan Kushnir, and Ronald R. Coifman. Texture separation via a reference set. *Applied and Computational Harmonic Analysis*, 2014.

[10] HM Krumholz, Z Lin, PS Keenan, and et al. Relationship between hospital readmission and mortality rates for patients hospitalized with acute myocardial infarction, heart failure, or pneumonia. *JAMA*, 2013.

[11] Prasanta Chandra Mahalanobis. On the generalized distance in statistics. *Proceedings of the National Institute of Sciences*, 1936.

[12] Gregorie Montavon, Mikio Braun, and Klaus-Robert Muller. Kernel analysis of deep networks. *Journal of Machine Learning Research*, 2011.

[13] R. Rojas. *Neural networks: a systematic introduction*. Springer Science and Business Media, 1996.

[14] Will E. Leeb Ronald R. Coifman. Earth mover's distance and equivalent metrics for spaces with hierarchical partition trees. *Yale CS Techinical Report*, 2013.

[15] L. Saul and S. Roweis. Think globally, fit locally: unsupervised learning of nonlinear manifolds. *Journal of Machine Learning Research*, 4(12):119–155, 2003.

[16] Amit Singer and Ronald R Coifman. Non-linear independent component analysis with diffusion maps. *Applied and Computational Harmonic Analysis*, 25(2):226–239, 2008.