Deep Computational Phenotyping

Zhengping Che*†, David Kale*†‡, Wenzhe Li†, Mohammad Taha Bahadori†, and Yan Liu†
†University of Southern California, Los Angeles, CA 90089
‡Whittier Virtual PICU, Children's Hospital Los Angeles, Los Angeles, CA 90027
{zche,dkale,wenzheli,mohammab,yanliu.cs}@usc.edu

ABSTRACT

We apply deep learning to the problem of discovery and detection of characteristic patterns of physiology in clinical time series data. We propose two novel modifications to standard neural net training that address challenges and exploit properties that are peculiar, if not exclusive, to medical data. First, we examine a general framework for using prior knowledge to regularize parameters in the topmost layers. This framework can leverage priors of any form, ranging from formal ontologies (e.g., ICD9 codes) to data-derived similarity. Second, we describe a scalable procedure for training a collection of neural networks of different sizes but with partially shared architectures. Both of these innovations are well-suited to medical applications, where available data are not yet Internet scale and have many sparse outputs (e.g., rare diagnoses) but which have exploitable structure (e.g., temporal order and relationships between labels). However, both techniques are sufficiently general to be applied to other problems and domains. We demonstrate the empirical efficacy of both techniques on two real-world hospital data sets and show that the resulting neural nets learn interpretable and clinically relevant features.

Categories and Subject Descriptors

J.3 [**Life and Medical Sciences**]: Health; I.5.1 [**Pattern Recognition**]: ModelsNeural nets

Keywords

Medical informatics; Phenotyping; Deep learning; Multivariate time series; Healthcare; Multi-label classification

1. INTRODUCTION

The increasing volume and availability of stored digital health data offers an unprecedented opportunity to improve future care by learning from past patient encounters. One important step towards this goal is learning richer, data-driven descriptions of illness. This field of research, known as *computational phenotyping*, has attracted many machine learning and data mining researchers, applying a broad set of methods to learn meaningful representations from a variety of data sources and types [39, 42, 17]. Most work on phenotyping from clinical time series has focused on variations of Gaussian processes and related models [27, 22] and focus on discovering cluster structure (e.g., disease subtypes).

Learning robust representations of human physiology is especially challenging because the underlying causes of health and wellness span body systems and physiologic processes, creating complex and nonlinear relationships among observed measurements (e.g., patients with septic shock may exhibit fever *or* hypothermia). Whereas classic shallow models (e.g., cluster models) may struggle in such settings, properly trained *deep neural networks* can often discover, model, and disentangle these types of latent factors [5] and extract meaningful abstract concepts from simple data types [20]. Because of these properties, deep learning has achieved state of the art results in speech recognition [15] and computer vision [38] and seems well-suited to computational phenotyping.

Recent work has demonstrated the potential of deep learning to derive insight from clinical data [21, 19, 37]. Nonetheless, the practical reality of neural networks remains challenging, and we face many open questions when applying them to a new problem. For example, do we have enough data? Deep learning's success is often associated with massive data sets of millions of examples [32, 11], but in medicine "big data" often means an Electronic Health Records (EHRs) database [14, 22] with only tens of thousands of cases. Other questions regard data preprocessing, model architecture, training procedures, etc. Answering these questions often requires time-consuming trial and error.

In addition, there is a wealth of existing medical knowledge that can inform analytics. Medicine is replete with ontologies, including SNOMED-CT [10], UMLS [9], and ICD-9 [23], whose structured forms are readily exploited by computational methods. Such ontologies have proven very useful for search, data mining, and decision support [13, 1]. For machine learning, such resources represent a source of potentially useful biases that can be used to accelerate learning. Combining structured knowledge with data-driven methods like deep learning presents a major challenge but also a significant opportunity for medical data mining.

In this paper we explore and propose solutions to some of the challenges that researchers face when utilizing deep learning to discover and detect significant physiologic pat-

*Authors made equal contributions.

KDD'15, August 10-13, 2015, Sydney, NSW, Australia.

DOI: http://dx.doi.org/10.1145/2783258.2783365.

terns in critically ill patients. By exploiting unique properties of both our domain (e.g., ontologies) and our data (e.g., temporal order in time series), we can improve the performance of our neural networks and make the training process more efficient. What is more, we show that each part of our framework addresses more general research challenges in deep learning. Our main contributions are as follows:

- We formulate a prior-based regularization framework for guiding the training of multi-label neural networks using medical ontologies and other structured knowledge. Our formulation is based on *graph Laplacian* priors [36, 40, 2, 3], which can represent any graph structure and incorporate arbitrary relational information. We apply graph Laplacian priors to the problem of training neural networks to classify physiologic time series with diagnostic labels, where there are many labels and severe class imbalance. What is more, this framework is general enough to incorporate data-driven (e.g., comorbidity patterns) and hybrid priors.

- We propose an efficient incremental training procedure for building a series of neural networks that detect physiologic patterns of increasing length. We use the parameters of an existing neural net to initialize the training of a new neural net designed to detect longer temporal patterns. This technique exploits both the well-known low rank structure of neural network weight matrices [12] and structure in our data domain, including temporal smoothness.

- We apply modern causal inference techniques [18] to the problem of analyzing and interpreting hidden layer representations of deep neural networks. We show empirically that these tools hold a great deal of promise for understanding the behavior of complex deep models and for helping to disentangle distributed representations. We use these methods to identify features that are most strongly associated with different diagnoses and to show that they often capture known pathophysiologies in critically ill patients.

We demonstrate the empirical efficacy of our deep learning framework using two real world clinical time series data sets. We show that our prior-based regularization framework improves performance on a very challenging multi-label classification task (predicting ICD-9 diagnostic codes) and that it is beneficial to incorporate both domain knowledge and data-driven similarity. We demonstrate that our incremental training procedure leads to faster convergence during training and learns features that are useful for classification and competitive with classically trained neural nets. We provide a thorough analysis of learned features that shows that the neural nets learn patterns that are interpretable and often associated with known critical illnesses. We finish by discussing the potential of this line of work for advancing research in both medical informatics and deep learning.

2. RELATED WORK

There is a growing body of work on computational phenotyping [42, 17], some of it using deep learning. [21] describes one of the first applications of modern deep learning to clinical time series. The authors use autoencoders to learn features from longitudinal clinical measurement time series and show that these features are both interpretable and useful for clustering and classifying patients. They focus on a narrow set of diseases and do not experiment with windows of different sizes or use supervised finetuning to learn more discriminative features. Although [22] do not use deep learning, they share similar goals: discovering meaningful physiologic patterns (or *physiomes*) in multivariate clinical time series. They formulate this as a clustering problem and apply a Gaussian mixture model. The resulting clusters are correlated with outcomes and known critical illnesses.

The application of deep learning to other sequential and temporal data provides a useful context for our work. Deep learning approaches have achieved breakthrough results in speech recognition [15]. We expect similar results (with time and effort) in more general time series data domains, including health. In natural language processing, distributed representations of words, learned from context using neural networks, have provided huge boosts in performance [33]. Our use of neural networks to learn representations of time series is similar: a window of time series observations can be viewed as the context for a single observation within that window.

Classic learning theory results show that the amount of training data required increases as the complexity of the learning algorithm increases [34, 4]. Thus, the flexibility of neural networks poses a challenge outside of traditional domains with access to massive Internet-scale data sets. This is especially true for medical applications where many predictive tasks suffer from severe class imbalance since most conditions are inherently. One possible remedy is to use *side information*, such as class hierarchy, as a rich prior to prevent overfitting and improve performance. However,there is still limited work in the deep learning community exploring the utility of such priors for training neural networks. To the best of our knowledge, [31] is the first work that combines a deep architecture with a tree-based prior to encode relations among different labels and label categories. Nonetheless, this work is limited to modeling a restricted class of side information.

Our work also has clear and interesting connections to ongoing research into efficient methods for training deep architectures. The renaissance of neural networks was launched by *unsupervised pretraining* [16, 6, 25]. The classic pretraining procedure can be viewed as a simple greedy method for building a deep architecture *vertically*, one layer at a time. Our incremental training method can be viewed as a greedy method for building deep architectures horizontally by adding units to one or more layers.

Our incremental training framework is also connected to two recent papers: [41] describe an incremental approach to feature learning in an online setting. They use a two step process to train new features: first, they train *only* the weights of the new features using a subset of training samples; then they retrain all weights on the full data. This approach outperforms fixed neural networks in streaming settings where the data and label distributions drift. There is an obvious parallel with our work, but we focus on changing input size, rather than data drift. Also, they do not analyze the convergence properties of their training procedure. [12] describe an approach for *predicting* parameters of neural networks by exploiting the smoothness of input data and the low rank structure of weight matrices. They decompose each weight matrix into a product of two low rank

matrices. One represents the learned weights for a subset of parameters. The other is a kernel similarity matrix, either designed using domain knowledge or estimated from data (using, e.g., covariance). In this way, parameter learning becomes a kernel regression problem. We use a related idea in our parameter initialization scheme: we exploit similarity between new inputs and old inputs to estimate initial parameter values prior to training.

3. METHODOLOGY

In this section, we describe our framework for performing effective deep learning on clinical time series data. We begin by discussing the Laplacian graph-based prior framework that we use to perform regularization when training multi-label neural networks. This allows us to effectively train neural networks, even with smaller data sets, and to exploit structured domain knowledge, such as ontologies. We then describe our incremental neural network procedure, which we developed in order to rapidly train a collection of neural networks to detect physiologic patterns of increasing length.

3.1 General Framework

Given a multivariate time series with P variables and length T, we can represent it as a matrix $\mathbf{X} \in \mathbb{R}^{P \times T}$. A *feature map* for time series $\mathbf{X}$ is a function $g : \mathbb{R}^{P \times T} \mapsto \mathbb{R}^D$ that maps $\mathbf{X}$ to a vector of features $\mathbf{x} \in \mathbb{R}^D$ useful for machine learning tasks like classification, segmentation, and indexing. Given the recent successes of deep learning, it is natural to investigate its effectiveness for feature learning in clinical time series data.

Suppose we have a data set of N multivariate time series, each with P variables and K binary labels. Without loss of generality, we assume all time series have the same length T. After a simple mapping that stacks all T column vectors in $\mathbf{X}$ to one vector $\mathbf{x}$, we have N labeled instances $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^N$, where $\mathbf{x}_i \in \mathbb{R}^D, \mathbf{y}_i \in \{0,1\}^K, D = PT$. The goal of multi-label classification is to learn a function f which can be used to assign a *set* of labels to each instance $\mathbf{x}_i$ such that $y_{ij} = 1$ if jth label is assigned to the instance $\mathbf{x}_i$ and 0 otherwise.

We use a deep feed-forward neural network, as shown in **Figure 1a**, with L hidden layers and an output prediction layer. We use $\mathbf{\Theta} = (\mathbf{\Theta}_{hid}, \boldsymbol{B})$ to denote the model parameters. $\mathbf{\Theta}_{hid} = \{\langle \boldsymbol{W}^{(\ell)}, \boldsymbol{b}^{(\ell)} \rangle\}_{\ell=1}^L$ denotes the weights for the hidden layers (each with $D^{(\ell)}$ units), and the K columns $\boldsymbol{\beta}_k \in \mathbb{R}^{D^{(L)}}$ of $\boldsymbol{B} = [\boldsymbol{\beta}_1 \boldsymbol{\beta}_2 \cdots \boldsymbol{\beta}_K]$ are the prediction parameters. For convenience we denote $\mathbf{h}^{(0)} = \mathbf{x}$ and $D^{(0)} = D$.

Throughout this paper, we assume a neural network with fully connected layers, linear activation $(\boldsymbol{W}^{(\ell)}\mathbf{h}^{(\ell-1)} + \boldsymbol{b}^{(\ell)})$ and sigmoid nonlinearities $(\sigma(z) = 1/(1 + \exp\{-z\}))$. Because of the modest size of our data, we pretrain each hidden layer as a denoising autoencoder (DAE) [35] by minimizing the reconstruction loss using stochastic gradient descent. In the supervised training stage, without any regularization, we treat multi-label classification as K separate logistic regressions, so the neural net has K sigmoid output units. To simplify the notation, let $\mathbf{h}_i = \mathbf{h}_i^{(L)} \in \mathbb{R}^{D^{(L)}}$ denote the output of top hidden layer for each instance $\mathbf{x}_i$. The conditional likelihood of $\mathbf{y}_i$ given $\mathbf{x}_i$ and model parameters $\mathbf{\Theta}$ can be written as:

$$\log p(\mathbf{y}_i|\mathbf{x}_i, \mathbf{\Theta}) = \sum_{k=1}^{K} \Big[y_{ik} \log \sigma(\boldsymbol{\beta}_k^\top \mathbf{h}_i) + (1 - y_{ik}) \log(1 - \sigma(\boldsymbol{\beta}_k^\top \mathbf{h}_i)) \Big]$$

Our framework can easily be extended to other network architectures, hidden unit types, and training procedures.

3.2 Prior-based regularization

Deep neural networks are known to work best in big data scenarios with many training examples. When we have access to only a few examples of each class label, incorporating prior knowledge can improve learning. [31] utilize tree-based prior, constructed from a hierarchy over image labels, to improve classification performance for smaller data sets with rare labels. However, the tree-based prior can only model a very restricted class of side information. In practice, we might have other types of prior information as well, such as pairwise similarity or co-occurrence. Thus, it is useful to have a more general framework able to incorporate a wider range of prior information in a unified way. Graph Laplacian-based regularization [36, 40, 2, 3] provides one such framework and is able to incorporate any relational information that can be represented as a (weighted) graph, including the tree-based prior as a special case.

Given a matrix $\mathbf{A} \in \mathbb{R}^{K \times K}$ representing pairwise connections or similarities, the Laplacian matrix is defined as $\mathbf{L} = \mathbf{C} - \mathbf{A}$, where $\mathbf{C}$ is a diagonal matrix with kth diagonal element $\mathrm{C}_{k,k} = \sum_{k'=1}^{K} (\mathrm{A}_{k,k'})$. $\mathbf{L}$ has the following property that makes it interesting for regularization. Given a set of K vectors vector of parameters $\boldsymbol{\beta}_k \in \mathbb{R}^{D^{(L)}}$ and :

$$\mathrm{tr}(\boldsymbol{\beta}^\top \mathbf{L} \boldsymbol{\beta}) = \frac{1}{2} \sum_{1 \le k,k' \le K} \mathrm{A}_{k,k'} \|\boldsymbol{\beta}_k - \boldsymbol{\beta}_{k'}\|_2^2, \qquad (1)$$

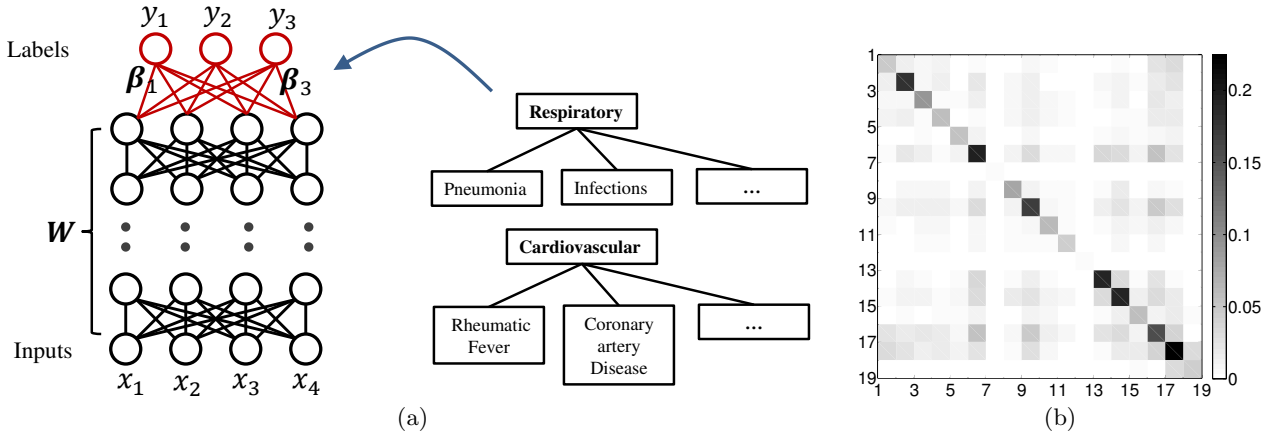
where $\mathrm{tr}(\cdot)$ represents the *trace* operator. According to **Equation 1**, the graph Laplacian regularizer enforces the parameters $\boldsymbol{\beta}_k$ and $\boldsymbol{\beta}_{k'}$ to be similar, proportional to $A_{k,k'}$. The Laplacian regularizer can be combined with other regularizers $R(\mathbf{\Theta})$ (e.g., the Frobenius norm $\|\boldsymbol{W}^{(\ell)}\|_F^2$ to keep hidden layer weights small), yielding the regularized loss function:

$$\mathcal{L} = -\sum_{i=1}^{N} \log p(\mathbf{y}_i|\mathbf{x}_i, \mathbf{\Theta}) + \lambda R(\mathbf{\Theta}) + \frac{\rho}{2} \mathrm{tr}(\boldsymbol{\beta}^\top \mathbf{L} \boldsymbol{\beta})$$

where $\rho, \lambda > 0$ are the Laplacian and other regularization hyperparameters, respectively. Note that the graph Laplacian regularizer is a quadratic in terms of parameters and so does not add significantly to the computational cost.

The graph Laplacian regularizer can represent any pairwise relationships between parameters. Here we discuss how to use different types of priors and the corresponding Laplacian regularizers to incorporate both structured domain knowledge (e.g., label hierarchies based on medical ontologies) and empirical similarities.

Structured domain knowledge as a tree-based prior. The graph Laplacian regularizer can represent a tree-based prior based on hierarchical relationships found in medical ontologies. In our experiments, we use diagnostic codes from the Ninth Revision of the *International Classification of Diseases* (ICD-9) system, which are widely used for classifying diseases and coding hospital data. The three digits (and two



(a) (b)

Figure 1: (a) A miniature illustration of the deep network with the regularization on categorical structure. The regularization is applied to the output layer of the network. (b) The co-occurrence matrix in the ICU dataset.

optional decimal digits) in each code form a natural hierarchy including broad body system categories (e.g., *Respiratory*), individual diseases (e.g., *Pneumonia*), and subtypes (e.g., *viral* vs. *Pneumococcal* pneumonia). **Figure 1a** illustrates two levels of the hierarchical structure of the ICD-9 codes. When using ICD-9 codes as labels, we can treat their ontological structure as prior knowledge. If two diseases belong to the same category, then we add an edge between them in the adjacency graph $\mathbf{A}$.

Data-driven similarity as a prior. Laplacian regularization is not limited prior knowledge in the form of trees or ontologies. It can also incorporate empirical priors, in the form of similarity matrices, estimated from data. For example, we can use the *co-occurrence* matrix $\mathbf{A} \in \mathbb{R}^{K \times K}$ whose elements are defined as follows:

$$\mathrm{A}_{k,k'} = \frac{1}{N} \sum_{i=1}^{N} \mathcal{I}(y_{ik} y_{ik'} = 1)$$

where N is the total number of the training data points, and $\mathcal{I}(\cdot)$ is the indicator function. Given the fact that $\mathrm{A}_{k,k'}$ is the maximum likelihood estimation of the joint probability $\mathbb{P}\{y_{ik} = 1, y_{ik'} = 1\}$, regularization with the Laplacian constructed from the co-occurrence similarity matrix encourages the learning algorithm to find a solution for the deep network that predicts the pair-wise joint probability of the labels accurately. The co-occurrence similarity matrix of 19 categories in ICU dataset is shown in **Figure 1b**.

3.3 Incremental training

Next we describe our algorithm for efficiently training a series of deep models to discover and detect physiologic patterns of varying lengths. This framework utilizes a simple and robust strategy for incremental learning of larger neural networks from smaller ones by iteratively adding new units to one or more layers. Our strategy is founded upon intelligent initialization of the larger network's parameters using those of the smaller network.

Given a multivariate time series $\mathbf{X} \in \mathbb{R}^{P \times T}$, there are two ways in which to use feature maps of varying or increasing lengths. The first would be to perform time series classification in an online setting in which we want to regularly re-classify a time series based on all available data. For example, we might would want to re-classify (or diagnose) a patient after each new observation while also including all previous data. Second, we can apply a feature map g designed for a shorter time series of length T_S to a longer time series of length $T > T_S$ using the *sliding window* approach: we apply g as a filter to subsequences of size T_S with stride R_S (there will be $\frac{T-T_S+1}{R_S}$). Proper choice of window size T_S and stride R_S is critical for producing effective features. However, there is often no way to choose the right T_S and R_S beforehand without *a priori* knowledge (often unavailable). What is more, in many applications, we are interested in multiple tasks (e.g., patient diagnosis *and* risk quantification), for which different values of T_S and R_S may work best. Thus, generating and testing features for many T_S and R_S is useful and often necessary. Doing this with neural nets can be computationally expensive and time-consuming.

To address this, we propose an incremental training procedure that leverages a neural net trained on windows of size T_S to initialize and accelerate the training of a new neural net that detects patterns of length $T' = T_S + \Delta T_S$ (i.e., ΔT_S additional time steps). That is, the input size of the first layer changes from $D = PT_S$ to $D' = D + d = PT_S + P\Delta T_S$.

Suppose that the existing and new networks have $D^{(1)}$ and $D^{(1)} + d^{(1)}$ hidden units in their first hidden layers, respectively. Recall that we compute the activations in our first hidden layer according to the formula $\mathbf{h}^{(1)} = \sigma(\boldsymbol{W}^{(1)}\mathbf{x} + \boldsymbol{b}^{(1)})$. This makes $\boldsymbol{W}^{(1)}$ an $D^{(1)} \times D$ matrix and $\boldsymbol{b}^{(1)}$ an $D^{(1)}$-vector; we have a row for each feature (hidden unit) in $\mathbf{h}^{(1)}$ and a column for each input in $\mathbf{x}$. From here on, we will treat the bias $\boldsymbol{b}^{(1)}$ as a column in $\boldsymbol{W}^{(1)}$ corresponding to a constant input and omit it from our notation.

The larger neural network has a $(D^{(1)} + d^{(1)}) \times (D + d)$ weight matrix $\boldsymbol{W}'^{(1)}$. The first D columns of $\boldsymbol{W}'^{(1)}$ correspond exactly to the D columns of $\boldsymbol{W}^{(1)}$ because they take the same D inputs. In time series data, these inputs are the observations in the same $T_S \times P$ matrix. We cannot guarantee the same identity for the first $D^{(1)}$ columns of $\boldsymbol{W}'^{(1)}$, which are the first $D^{(1)}$ hidden units of $\mathbf{h}'^{(1)}$; nonetheless, we can make a reasonable assumption that these hidden units are highly similar to $\mathbf{h}^{(1)}$. Thus, we can think of constructing $\boldsymbol{W}'^{(1)}$ by adding d new columns and $d^{(1)}$ new rows to

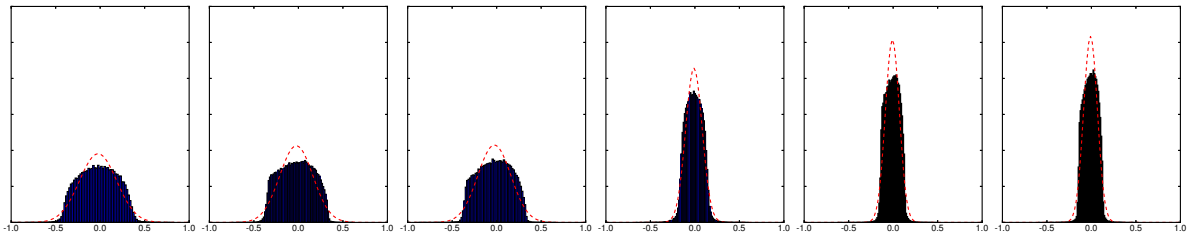


Figure 2: Weight distributions for three layers of a neural network after pretraining (*left three*) and finetuning (*right three*).

Algorithm 1 Similarity-based Initialization

Input: Training data $\mathbf{X} \in \mathbb{R}^{N \times (D+d)}$; existing weights $\boldsymbol{W}^{(1)} \in \mathbb{R}^{D^{(1)} \times D}$; kernel function $\mathbf{k}(\cdot, \cdot)$
Output: Initialized weights $\Delta \boldsymbol{W}_{ne} \in \mathbb{R}^{D^{(1)} \times d}$

for each new input dimension $i \in [1, d]$ **do**
  for each existing input dimension $k \in [1, D]$ **do**
    Let $\mathrm{K}[D+i, k] := \mathbf{k}(\mathrm{X}[\cdot, D+i], \mathrm{X}[\cdot, k])$
  end for
  Normalize $\mathbf{K}$ (if necessary)
  for each existing feature $j \in [1, D^{(1)}]$ **do**
    Let $\Delta W_{ne}[j, i] := \sum_{k=1}^{D} \mathrm{K}[D+i, k] W^{(1)}[j, k]$
  end for
end for

$\boldsymbol{W}^{(1)}$. As illustrated in **Figure 3**, the new weights can be divided into three categories.

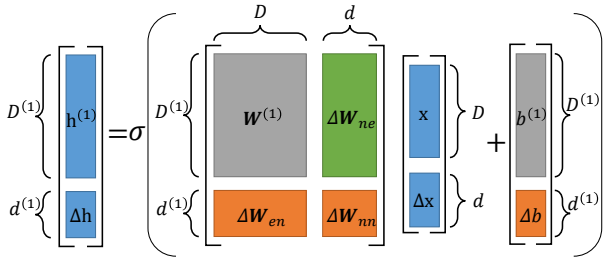


Figure 3: How adding various units changes the weights $\boldsymbol{W}$.

- $\Delta \boldsymbol{W}_{ne}$: $D^{(1)} \times d$ weights that connect new inputs to existing features.
- $\Delta \boldsymbol{W}_{en}$: $d^{(1)} \times D$ weights that connect existing inputs to new features.
- $\Delta \boldsymbol{W}_{nn}$: $d^{(1)} \times d$ weights that connect new inputs to new features.

We now describe strategies for using $\boldsymbol{W}^{(1)}$ to choose initial values for parameters in each category.

Similarity-based initialization for new inputs To initialize $\Delta \boldsymbol{W}_{ne}$, we leverage the fact that we can compute or estimate the similarity among inputs. Let $\mathbf{K}$ be a $(D+d) \times (D+d)$ kernel similarity matrix between the inputs to the larger neural network that we want to learn. We can estimate the weight between the ith new input (i.e., input $D+i$) and the jth hidden unit as a linear combination of the parameters for the existing inputs, weighted by each existing input's similarity to the ith new input. This is shown in **Algorithm 1**.

Algorithm 2 Gaussian Sampling-based Initialization

Input: Existing weights $\boldsymbol{W}^{(1)} \in \mathbb{R}^{D^{(1)} \times D}$
Output: Initialized weights $\Delta \boldsymbol{W}_{en} \in \mathbb{R}^{d^{(1)} \times D}$, $\Delta \boldsymbol{W}_{nn} \in \mathbb{R}^{d^{(1)} \times d}$

Let $\bar{w} = \frac{1}{DD^{(1)}} \sum_{i,j} W^{(1)}[i, j]$
Let $\bar{s} = \frac{1}{DD^{(1)}-1} \sum_{i,j} (W^{(1)}[i, j] - \bar{w})^2$
for each new feature $j \in [1, d^{(1)}]$ **do**
  for each existing input dimension $i \in [1, D]$ **do**
    Sample $\Delta W_{ne}[j, i] \sim \mathcal{N}(\bar{w}, \bar{s})$
  end for
  for each new input dimension $i \in [1, d]$ **do**
    Sample $\Delta W_{nn}[j, i] \sim \mathcal{N}(\bar{w}, \bar{s})$
  end for
end for

Choice of $\mathbf{K}$ is a matter of preference and input type. A time series-specific similarity measure might assign a zero for each pair of inputs that represents different variables (i.e., different univariate time series) and otherwise emphasize temporal proximity using, e.g., a squared exponential kernel. A more general approach might estimate similarity empirically, using sample covariance or cosine similarity. We find that the latter works well, for both time series inputs and arbitrary hidden layers.

Sampling-based initialization for new features When initializing the weights for $\boldsymbol{W}_{en}$, we do not have the similarity structure to guide us, but the weights in $\boldsymbol{W}^{(1)}$ provide information. A simple but reasonable strategy is to sample random weights from the empirical distribution of entries in $\boldsymbol{W}^{(1)}$. We have several choices here. The first regards whether to assume and estimate a parametric distribution (e.g., fit a Gaussian) or use a nonparametric approach, such as a kernel density estimator or histogram. The second regards whether to consider a single distribution over all weights or a separate distribution for each input.

In our experiments, we found that the existing weights often had recognizable distributions (e.g., Gaussian, see **Figure 2**) and that it was simplest to estimate and sample from a parametric distribution. We also found that using a single distribution over all weights worked as well as, if not better than, a separate distribution for each input.

For initializing weights in $\boldsymbol{W}_{nn}$, which connect new inputs to new features, we could apply either strategy, as long as we have already initialized $\boldsymbol{W}_{en}$ and $\boldsymbol{W}_{ne}$. We found that estimating all new feature weights (for existing or new inputs) from the same simple distribution (based on $\boldsymbol{W}^{(1)}$) worked

Table 1: AUROC for classification.

	Tasks	*No Prior*	*Co-Occurrence*	*ICD-9 Tree*
Subsequence	*All*	0.7079 ± 0.0089	$\mathbf{0.7169 \pm 0.0087}$	0.7143 ± 0.0066
	Categories	0.6758 ± 0.0078	$\mathbf{0.6804 \pm 0.0109}$	0.6710 ± 0.0070
	Labels	0.7148 ± 0.0114	$\mathbf{0.7241 \pm 0.0093}$	0.7237 ± 0.0081
Episode	*All*	0.7245 ± 0.0077	$\mathbf{0.7348 \pm 0.0064}$	0.7316 ± 0.0062
	Categories	0.6952 ± 0.0106	$\mathbf{0.7010 \pm 0.0136}$	0.6902 ± 0.0118
	Labels	0.7308 ± 0.0099	$\mathbf{0.7414 \pm 0.0064}$	0.7407 ± 0.0070

best. Our full Gaussian sampling initialization strategy is shown in **Algorithm 2**.

Initializing other layers This framework generalizes beyond the input and first layers. Adding d' new hidden units to $\mathbf{h}'^{(1)}$ is equivalent to adding d' new inputs to $\mathbf{h}'^{(2)}$. If we compute the activations in $\mathbf{h}'^{(1)}$ for a given data set, these become the new inputs for $\mathbf{h}'^{(2)}$ and we can apply both the similarity and sampling-based strategies to initialize new entries in the expanded weight matrix $\boldsymbol{W}'^{(2)}$. The same goes for all layers. While we can no longer design special similarity matrices to exploit known structure in the inputs, we can still estimate empirical similarity from training data activations in, e.g., $\mathbf{h}'^{(2)}$.

Intuition suggests that if our initializations from the previous pretrained values are sufficiently good, we may be able to forego pretraining and simply perform backpropagation. Thus, we choose to initialize with pretrained weights, then do the supervised finetuning on all weights.

4. EXPERIMENTS

To evaluate our framework, we ran a series of classification and feature-learning experiments using two collections of clinical time series collected during the delivery of care in intensive care units (ICUs) at large hospitals. In **Section 4.1**, we demonstrate the benefit of using priors (both knowledge- and data-driven) to regularize the training of multi-label neural nets. In **Section 4.2**, we show that incremental training both speeds up training of larger neural networks and keeps classification performance. In **Section 4.3**, we perform a qualitative analysis of the learned features, showing that neural nets can learn clinically significant physiologic patterns. Finally, in **Section 4.4** we show that priors can help to disentangle the latent factors of variation modeled in the upper layer of the neural net.

Physionet Challenge 2012 Data. The first data set comes from *PhysioNet Challenge 2012* website [30] which is a publicly available[1] collection of multivariate clinical time series from 8000 ICU units. Each episode is a multivariate time series of roughly 48 hours and containing over 30 variables. These data come from one ICU and four specialty units, including coronary care and cardiac and general surgery recovery units. We use the *Training Set A* subset for which outcomes, including in-hospital mortality, are available. We resample the time series on an hourly basis and propagate measurements forward (or backward) in time to fill gaps. We scale each variable to fall between $[0, 1]$. We discuss handling of entirely missing time series below.

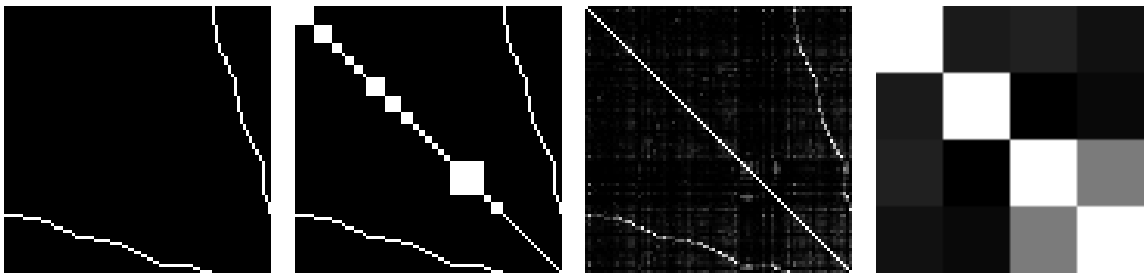
[1] http://physionet.org/challenge/2012/

ICU Data. The second data set consists of ICU clinical time series extracted from the electronic health records (EHRs) system of a major hospital. The original data set includes roughly ten thousand episodes of varying lengths, but we exclude episodes shorter than 12 hours or longer than 128 hours, yielding a data set of 8500 multivariate time series of a dozen physiologic variables, which we resample once per hour and scale to [0,1]. Each episode has zero or more associated diagnostic codes from the Ninth Revision of the *International Classification of Diseases* (ICD-9). From the raw 3-5 digit ICD-9 codes, we create a two level hierarchy of labels and label categories using a two-step process. First, we truncate each code to the tens position (with some special cases handled separately), thereby merging related diagnoses and reducing the number of unique labels. Second, we treat the standard seventeen broad groups of codes (e.g., 460-519 for respiratory diseases), plus the supplementary V and E groups as label categories. After excluding one category that is absent in our data, we have 67 unique labels and 19 categories.

We implemented all neural networks in Theano [8] as variations of a multilayer perceptron with five hidden layers (of the same size) of sigmoid units. The input layer has PT input units for P variables and T time steps, while the output layer has one sigmoid output unit per label. Except when we use our incremental training procedure, we initialize each neural network by training it as an unsupervised stacked denoising autoencoder (SDAE). We found this helps significantly because our data sets are relatively small and our labels are quite sparse.

4.1 Benefits of prior-based regularization

Our first set of experiments demonstrates the utility of using priors to regularize the training of multi-label neural networks, especially when labels are sparse and highly correlated or similar. From each time series, we extract all subsequences of length $T = 12$ in sliding window fashion, with an overlap of 50% (i.e., stride $R = 0.5T$), and each subsequence receives its episode's labels (e.g., diagnostic code or outcome). We use these subsequences to train a single unsupervised SDAE with five layers and increasing levels of corruption (from 0.1 to 0.3), which we then use to initialize the weights for all supervised neural networks. The sparse multi-label nature of the data makes stratified k-fold cross validation difficult, so we instead randomly generate a series of 80/20 random training/test splits of episodes and keep the first five that have at least one positive example for each label or category. At testing time, we measure classification performance for both frames and episodes. We make episode-level predictions by thresholding the mean score for all subsequences from that episode.



(a) ICD-9 Tree. (b) ICD-9 shared category. (c) ICU co-occurrence. (d) Physionet co-occur.

Figure 4: Example priors for the ICU (a-c) and Physionet (d) data sets.

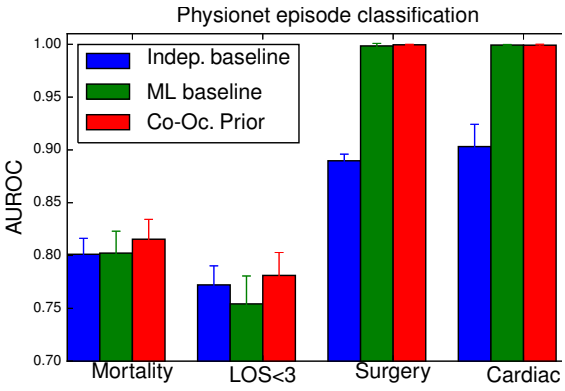


Figure 5: Physionet classification performance.

The ICU data set contains 8500 episodes varying in length from 12 to 128 hours. The above subsequence procedure produces 50,000 subsequences. We treat the simultaneous prediction of all 86 diagnostic labels and categories as a multi-label prediction problem. This lends itself naturally to a tree-based prior because of the hierarchical structure of the labels and categories (**Figure 4a, 4b**). However, we also test a data-based prior based on co-occurrence (**Figure 4c**). Each neural network has an input layer of 156 units and five hidden layers of 312 units each.

The Physionet data set contains 3940 episodes, most of length 48 hours, and yields 27,000 subsequences. These data have no such natural label structure to leverage, so we simply test whether a data-based prior can improve performance. We create a small multi-label classification problem consisting of four binary labels with strong correlations, so that similarity-based regularization should help: in-hospital mortality (*mortality*), length-of-stay less than 3 days (*los<3*), whether the patient had a cardiac condition (*cardiac*), and whether the patient was recovering from surgery (*surgery*). The mortality rate among patients with length-of-stay less than 3 days is nearly double the overall rate. The cardiac and surgery are created from a single original variable indicating which type of critical care unit the patient was admitted to; nearly 60% of cardiac patients had surgery. **Figure 4d** shows the co-occurrence similarity between the labels.

We impute missing time series (where a patient has no measurements of a variable) with the median value for patients in the same unit. This makes the cardiac and surgery prediction problems easier but serves to demonstrate the efficacy of our prior-based training framework. Each neural network has an input layer of 396 units and five hidden layers of 900 units each.

The results for Physionet are shown in **Figure 5**. We observe two trends, which both suggest that multi-label neural networks work well and that priors help. First, jointly learning features, even without regularization, can provide a significant benefit. Both multi-label neural networks dramatically improve performance for the *surgery* and *cardiac* tasks, which are strongly correlated and easy to detect because of our imputation procedure. In addition, the addition of the co-occurrence prior yields clear improvements in the *mortality* and *los<3* tasks while maintaining the high performance in the other two tasks. Note that this is *without tuning the regularization parameters.*

Table 1 shows the results for the ICU data set. We report classification AUROC performance for both individual subsequences and episodes, computed across all outputs, as well as broken down into just labels and just categories. The priors provide some benefit but the improvement is not nearly as dramatic as it is for Physionet. We face a rather extreme case of class imbalance (some labels have fewer than 0.1% positive examples) multiplied across dozens of labels. In such settings, predicting all negatives yields a very low loss. We believe that even the prior-based regularization suffers from the imbalanced classes: enforcing similar parameters for equally rare labels may cause the model to make few positive predictions. However, the Co-Occurrence prior does provide a clear benefit, even in comparison to the ICD-9 prior. As **Figure 4c** shows, this empirical prior captures not only the category/label relationship encoded by the ICD-9 tree prior but also includes valuable cross-category relationships that represent commonly co-morbid conditions.

4.2 Efficacy of incremental training

In these experiments we show that our incremental training procedure not only produces more effective classifiers (by allowing us to combine features of different lengths) but also speeds up training. We train a series of neural networks designed to model and detect patterns of lengths $T_S = 12, 16, 20, 24$. Each neural net has PT_S inputs (for P variables) and five layers of $2PT_S$ hidden units each. We use each neural network to make an episode-level prediction as before (i.e., the mean real-valued output for all frames)

and then combine those predictions to make a single episode level prediction. We combine two training strategies:
Full: separately train each neural net, with unsupervised pretraining followed by supervised finetuning.
Incremental: fully train the smallest ($T_S = 12$) neural net and then use its weights to initialize supervised training of the next model ($T_S = 16$). Repeat for subsequent networks.

We run experiments on a subset of the ICU data set, including only the 6200 episodes with at least 24 hours and no more than 128 hours of measurements. This data set yields 50000, 40000, 30000, and 20000 frames of lengths 12, 16, 20, and 24, respectively.

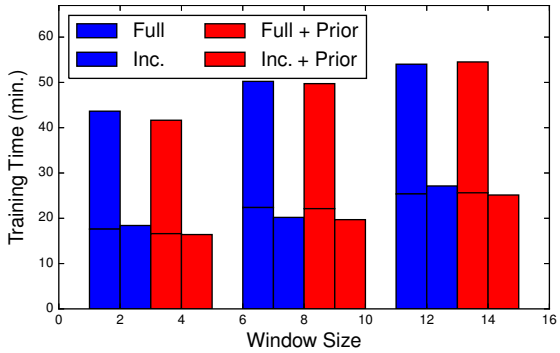


Figure 6: Training time for different neural networks for full/incremental training strategies.

Table 2: AUROC for incremental training.

Size	Level	*Full*	*Inc*	*Prior+Full*	*Prior+Inc*
16	Subseq.	0.6928	0.6874	0.6556	0.6581
	Episode	0.7148	0.7090	0.6668	0.6744
20	Subseq.	0.6853	0.6593	0.6674	0.6746
	Episode	0.7022	0.6720	0.6794	0.6944
24	Subseq.	0.7002	0.6969	0.6946	0.7008
	Episode	0.7185	0.7156	0.7136	0.7171

We begin by comparing the training time (in minutes) saved by incremental learning in **Figure 6**. Incremental training provides an alternative way to initialize larger neural networks and allows us to forego unsupervised pretraining. What is more, supervised finetuning converges just as quickly for the incrementally intialized networks as it does for the fully trained network. As a result, it reduces training time for a single neural net by half. **Table 2** shows the that the incremental training reaches comparable performance. Moreover, the combination of the incremental training and Laplacian prior leads to better performance than using Laplacian prior only.

4.3 Qualitative Analysis of Features

Figure 7 shows visualizations of two the significant physiologic patterns learned by the neural network with the ICD-9 Tree prior. In each case, we used a feature selection procedure to identify a subset of hidden units in the topmost hidden layer that are most strongly associated with a particular label or category. We then found the 50 input subsequences with the highest activations in those units and plotted the mean trajectories for 12 of the 13 physiologic variables. **Figure 7a** visualizes features that were found to be causally related to the circulatory disease category using the causal inference procedure described in the next section. We see these features detect highly elevated blood pressure and heart rate, as well as depressed pH. The features also detect elevated end-tidal CO2 (ETCO2) and fraction-inspired oxygen (FIO2), which likely indicate ventilation and severe critical illness. Interesting, these features also detect elevated urine output, and thus it is not surprising that these features are also correlated with labels related to urinary disorders. **Figure 7b** visualizes the patterns detected by features that are highly correlated with septic shock. Unsurprisingly, they detect very irregular physiology, including extremely low Glascow Coma Score (indicating the patient may be unconscious), as well as evidence of ventilation. These are all common symptoms of shock.

4.4 Causality Analysis

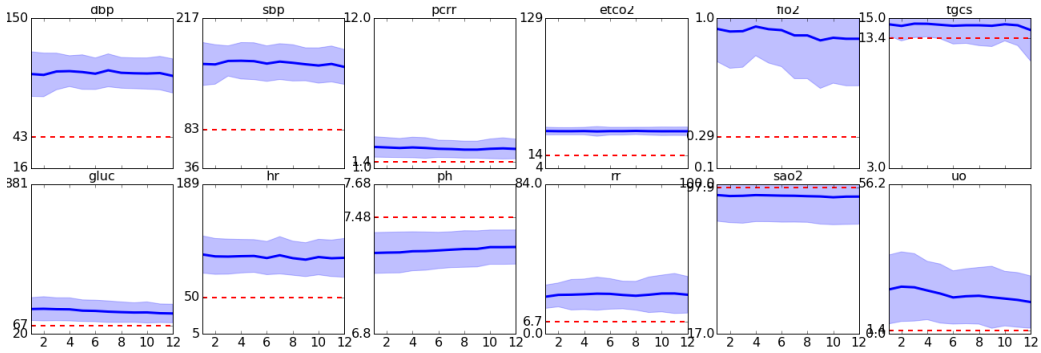
One of the main advantages of representation learning with deep networks is their ability to disentangle factors of variation present in the data but unobserved [5]. If the algorithm does a good job of disentangling the underlying factors, the subsequent learning will become much easier since it counteracts the curse of dimensionality [7]. In addition, knowledge about one factor usually can improve the estimation about another [26].

We apply tools from causal inference to investigate the ability of neural nets (with and without prior-based regularization) to disentangle factors of variation. This is a critical component of applying deep learning to medical data, since decisions about treatment and care can significantly impact patient lives and outcomes. Thus, discerning correlation from true causal relationships is of vital importance. To this end, we note that classic causal inference algorithms require a set of causal priors to be available for the variable to be able to cancel out the impact of spurious causation paths [24, Chapter 3]. While we do not have such priors available for our labels, we can still identify potential causation among the variables if they are not distributed according to Gaussian distribution [28, 29, 18]. Our labels are binary and hidden features logistic, and so they satisfy the requirements of these algorithms.

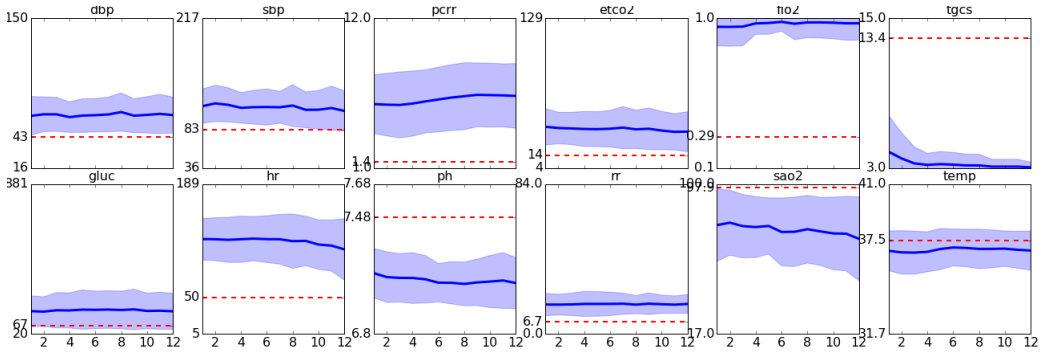
We use causal analysis both quantitatively and qualitatively. For quantitative analysis, we informally compare the strength of the detected causal relationships, as we do not have access to the ground truth causal relationships. We find the causal edges between each feature and label using the state-of-the-art *Pairwise LiNGAM* algorithm from [18]. In order to find the magnitude of the causal edges, according to the authors' recommendation we fit a logistic regression model to the features selected by Pairwise LiNGAM. The results in Table 3 show the Frobenius norm of the causal predictive weight matrix $\widehat{\boldsymbol{B}}$ learned for the ICU dataset. The results indicate a near 7% improvement when using the co-occurrence prior over the baseline neural net, suggesting that this prior may help disentangle latent factors in the data and make it easier to find potential causal relationships between learned features and outcomes.

5. CONCLUSION AND FUTURE WORK

The boom of digital health data from EHRs, wearable health devices, and other sources presents an unprecedented opportunity for future clinical research. Capitalizing on this opportunity requires bringing to bear modern computational



(a) Phenotypic patterns for ICD-9 circulatory disease category (390-459).



(b) Phenotypic patterns for conditions related to septic shock (ICD-9 codes 990-995).

Figure 7: Example features learned from the ICU data.

Table 3: Magnitude of the causal relationships identified by using the representations learned by the deep learning algorithm.

No Prior	*Co-occurrence*	*ICD-9 Tree*
252.61	**270.28**	242.50

techniques from data mining and machine learning, and in turn we must adapt these tools to solve problems unique to medicine. In this paper, we demonstrated a robust suite of tools for applying deep learning to the discovery and detection of significant physiology in hospital data. In doing so, we address challenges presented by clinical time series and exploit properties peculiar, if not unique, to these data.

First, we proposed using neural networks, trained on windows of multivariate clinical time series, to discover physiologic patterns associated with known clinical phenotypes and predictive of health outcomes. Our experiments showed that this simple approach can discover multivariate temporal patterns that are both predictive and interpretable.

Next we addressed the limited data and and sparse labels common in EHRs by leveraging existing domain knowledge (e.g., ontologies) as a prior during learning. We transform such knowledge into a graph Laplacian, which can be used as a simple, computationally efficient regularizer. Our experimental results showed that priors can improve classification performance and help to learn more clinically relevant features. This framework also generalizes to data-driven and other priors (e.g., empirical disease co-morbidity).

Then we proposed an efficient incremental learning framework for discovery and detection of different length patterns in clinical time series. Because longer temporal subsequences overlap shorter ones, neural nets trained on these patterns will have overlapping inputs and share structure in higher layers. Armed with this insight, we developed a simple but effective way to initialize a larger network's parameters using an existing smaller network. Our empirical results showed that this approach dramatically speeds up training without hurting classification.

Finally, we demonstrated how state-of-the-art causal inference algorithms can be used to help evaluate the physiologic representations learned by neural nets.

We believe that the ability to discover potentially meaningful patterns of physiology and other symptoms from these data via automated means will be an important tool for future clinical studies. Our results demonstrate the promise of learned representations for health "big data" problems. We have several next steps. The first is to apply our framework to a broader set of clinical tasks and data sets and to work with experts to validate our findings. We are also investigating how to account for the noisy, unreliable nature of diagnostic codes; one possibility is to use them for semi-supervised, rather than fully supervised, learning. Finally, we are very passionate about the idea of *causal feature learning*, in which the goal is to learn latent space representations that enable more robust causal reasoning.

6. ACKNOWLEDGMENTS

The authors would like to thank Randall Wetzel of Children's Hospital Los Angeles for his assistance in understanding and analyzing clinical data. David Kale was supported by the Alfred E. Mann Innovation in Engineering Doctoral Fellowship, and the VPICU was supported by grants from

the Laura P. and Leland K. Whitter Foundation. Mohammad Taha Bahadori was supported by NSF award number IIS-1254206. Yan Liu was supported by NSF IIS-1134990 and IIS-1254206 awards. The views and conclusions are those of the authors and should not be interpreted as representing the official policies of the funding agency, or the U.S. Government.

7. REFERENCES

[1] S. L. Achour, M. Dojat, C. Rieux, P. Bierling, and E. Lepage. A umls-based knowledge acquisition tool for rule-based clinical decision support system development. *JAMIA*, 8(4):351–360, 2001.

[2] R. K. Ando and T. Zhang. Learning on graph with laplacian regularization. *NIPS*, 2007.

[3] M. T. Bahadori, Q. R. Yu, and Y. Liu. Fast multivariate spatio-temporal analysis via low rank tensor learning. In *NIPS*, 2014.

[4] P. L. Bartlett and S. Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *JMLR*, 2003.

[5] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2013.

[6] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle. Greedy layer-wise training of deep networks. In *NIPS*. 2007.

[7] Y. Bengio, G. Mesnil, Y. Dauphin, and S. Rifai. Better mixing via deep representations. In *ICML*, 2013.

[8] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio. Theano: a CPU and GPU math expression compiler. In *SciPy*, 2010.

[9] O. Bodenreider. The unified medical language system (umls): integrating biomedical terminology. *Nucleic Acids Res.*, 2004.

[10] R. Cornet and N. de Keizer. Forty years of snomed: a literature review. *BMC medical informatics and decision making*, 8(Suppl 1):S2, 2008.

[11] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[12] M. Denil, B. Shakibi, L. Dinh, M. Ranzato, and N. de Freitas. Predicting parameters in deep learning. In *NIPS*. 2013.

[13] M. C. Díaz-Galiano, M. T. Martín-Valdivia, and L. Ureña-López. Query expansion with a medical ontology to improve a multimodal information retrieval system. *Comput. Biol. Med.*, 2009.

[14] A. Goldberger, L. N. Amaral, L. Glass, J. Hausdorff, P. Ivanov, R. Mark, J. Mietus, G. Moody, C. Peng, and H. Stanley. Physiobank, physiotoolkit, and physionet: Components of a new research resource for complex physiologic signals. *Circulation*, 2000.

[15] A. Graves and N. Jaitly. Towards end-to-end speech recognition with recurrent neural networks. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1764–1772, 2014.

[16] G. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural Comput.*, 2006.

[17] J. C. Ho, J. Ghosh, and J. Sun. Marble: high-throughput phenotyping from electronic health records via sparse nonnegative tensor factorization. In *KDD*, 2014.

[18] A. Hyvärinen and S. M. Smith. Pairwise likelihood ratios for estimation of non-gaussian structural equation models. *JMLR*, 2013.

[19] D. Kale, Z. Che, Y. Liu, and R. Wetzel. Computational discovery of physiomes in critically ill children using deep learning. In *DMMI Workshop, AMIA*, volume 2014.

[20] A. Karpathy and L. Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *CVPR*, 2015.

[21] T. A. Lasko, J. Denny, and M. Levy. Computational phenotype discovery using unsupervised feature learning over noisy, sparse, and irregular clinical data. *PLoS ONE*, 2013.

[22] B. Marlin, D. Kale, R. Khemani, and R. Wetzel. Unsupervised pattern discovery in electronic health care data using probabilistic clustering models. In *IHI*, 2012.

[23] W. H. Organization. *International statistical classification of diseases and related health problems*. 2004.

[24] J. Pearl. *Causality: models, reasoning and inference*. Cambridge Univ Press, 2009.

[25] M. Ranzato, C. Poultney, S. Chopra, and Y. L. Cun. Efficient learning of sparse representations with an energy-based model. In *NIPS*. 2007.

[26] S. Reed, K. Sohn, Y. Zhang, and H. Lee. Learning to disentangle factors of variation with manifold interaction. In *ICML*, 2014.

[27] P. Schulam, F. Wigley, and S. Saria. Clustering longitudinal clinical marker trajectories from electronic health data: Applications to phenotyping and endotype discovery. 2015.

[28] S. Shimizu, P. O. Hoyer, A. Hyvärinen, and A. Kerminen. A linear non-gaussian acyclic model for causal discovery. *JMLR*, 2006.

[29] S. Shimizu, T. Inazumi, Y. Sogawa, A. Hyvärinen, Y. Kawahara, T. Washio, P. O. Hoyer, and K. Bollen. Directlingam: A direct method for learning a linear non-gaussian structural equation model. *JMLR*, 2011.

[30] I. Silva, G. Moody, D. J. Scott, L. A. Celi, and R. G. Mark. Predicting in-hospital mortality of icu patients: The physionet/computing in cardiology challenge 2012. *Computing in cardiology*, 2012.

[31] N. Srivastava and R. R. Salakhutdinov. Discriminative transfer learning with tree-based priors. In *NIPS*, pages 2094–2102, 2013.

[32] A. Torralba, R. Fergus, and W. T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *PAMI*, 2008.

[33] J. Turian, L. Ratinov, and Y. Bengio. Word representations: A simple and general method for semi-supervised learning. In *ACL*, 2010.

[34] V. Vapnik. *The nature of statistical learning theory*. Springer Science & Business Media, 2000.

[35] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *ICML*, 2008.

[36] K. Q. Weinberger, F. Sha, Q. Zhu, and L. K. Saul. Graph laplacian regularization for large-scale semidefinite programming. In *NIPS*, 2006.

[37] G. Wu, M. Kim, Q. Wang, Y. Gao, S. Liao, and D. Shen. Unsupervised deep feature learning for deformable registration of mr brain images. In *MICCAI*. 2013.

[38] R. Wu, S. Yan, Y. Shan, Q. Dang, and G. Sun. Deep image: Scaling up image recognition. *arXiv:1501.02876*, 2015.

[39] T. Xiang, D. Ray, T. Lohrenz, P. Dayan, and P. R. Montague. Computational phenotyping of two-person interactions reveals differential neural response to depth-of-thought. *PLoS Comput. Biol.*, 2012.

[40] T. Zhang, A. Popescul, and B. Dom. Linear prediction models with graph regularization for web-page categorization. In *KDD*, 2006.

[41] G. Zhou, K. Sohn, and H. Lee. Online incremental feature learning with denoising autoencoders. In *AISTATS*, 2012.

[42] J. Zhou, F. Wang, J. Hu, and J. Ye. From micro to macro: Data driven phenotyping by densification of longitudinal electronic medical records. In *KDD*, 2014.