DDL: Deep Dictionary Learning for Predictive Phenotyping

Tianfan Fu[1,*], **Trong Nghia Hoang**[2,*], **Cao Xiao**[3] and **Jimeng Sun**[1]

[1] Department of Computational Science and Engineering, Georgia Institute of Technology
[2]MIT-IBM Watson AI Lab, IBM Research
[3] Analytics Center of Excellence, IQVIA
tfu42@gatech.edu, nghiaht@ibm.com, cao.xiao@iqvia.com, jsun@cc.gatech.edu

Abstract

Predictive phenotyping is about accurately predicting what phenotypes will occur in the next clinical visit based on longitudinal Electronic Health Record (EHR) data. While deep learning (DL) models have recently demonstrated strong performance in predictive phenotyping, they require access to a large amount of labeled data, which are expensive to acquire. To address this label-insufficient challenge, we propose a deep dictionary learning framework (DDL) for phenotyping, which utilizes unlabeled data as a complementary source of information to generate a better, more succinct data representation. Our empirical evaluations on multiple EHR datasets demonstrated that DDL outperforms the existing predictive phenotyping methods on a wide variety of clinical tasks that require patient phenotyping. The results also show that unlabeled data can be used to generate better data representation that helps improve DDL's phenotyping performance over existing methods that only uses labeled data.

1 Introduction

The recent rise in popularity of deep learning (DL) and the widespread use of electronic health record (EHR) data in clinical research have sparked strong interest in using DL for electronic phenotyping, which is of paramount importance in various health analytics tasks such as chronic disease diagnosis [Lipton *et al.*, 2015; Ma *et al.*, 2017], patient subtyping [Baytas *et al.*, 2017] and disease prediction [Esteban *et al.*, 2016]. The key advantage of DL models are their abilities to construct deep features that capture complex and long-range data dependencies efficiently, both of which are particular traits of biomedical data. This helps DL models achieve better performance and require less effort in feature engineering than classical machine learning (ML) models.

Despite these successes, the training of DL-based phenotyping models often relies on direct supervision via labeled data, which in turn requires access to a large volume of EHR data annotated with relevant medical labels. This highlights a key weakness of existing DL-based phenotyping models in terms of practical efficiency and data utilization. First, acquiring labeled data is often prohibitively expensive due to the incurred labor-intensive data annotation, which makes the training practically inefficient. Second, most DL models are trained using supervised learning methods, which do not make use of unlabeled EHR data [Lipton *et al.*, 2015; Choi *et al.*, 2016b; Ma *et al.*, 2017, 2018]. Ignoring unlabeled data, however, leads to a severe under-utilization of available information since such data can still be exploited to learn a better representation of labeled data (hence, better prediction performance) even though they do not contain predictive information. To mitigate the above weaknesses, this paper aims to develop a deep dictionary learning (DDL) framework that utilizes and combines both labeled and unlabeled data to learn a better data representation for predictive phenotyping.

To elaborate, DDL implements a hybrid learning architecture that combines ideas from both dictionary learning and recurrent neural network (RNN) for temporal prediction. This is achieved by first embedding raw, *unlabeled* EHR data into a latent feature space using DDL's RNN component. Then, dictionary learning is applied on the resulting set of embedded representations to search for a smaller set of patterns that occur frequently across different patients' representations. This in turn allows DDL to characterize each patient's embedded representation as a linear combination of those frequent patterns, which constitutes their *deep dictionary representations* (Section 2.2).

Furthermore, to optimize the above *deep dictionary representation*, DDL develops a *deep dictionary decoder* that reconstructs a patient's *unlabeled* EHR data from its deep dictionary representation. This is achieved by minimizing the decoder's reconstruction loss, which helps establish a parsimonious set of hidden-layer *patient prototypes* (Section 2.3). Different patients can then be viewed as different combinations of these prototypes where the combination coefficients represent its high-level feature description. These coefficients are then coupled with the patients' labeled data to learn a mapping from each patient's high-level description to his/her target outcomes (Section 2.4). This is also demonstrated in our experiments (Section 3) that such representation indeed induces better performance than those of the existing baselines for predictive phenotyping.

Last but not least, DDL also develops an end-to-end training

[*] Tianfan Fu and Trong Nghia Hoang contribute equally.

mechanism that optimizes the above data embedding (Section 2.2), reconstruction (Section 2.3) and prediction (Section 2.4) components simultaneously. This intuitively allows DDL's representation components (built with unlabeled data) to coordinate with its prediction component (built with labeled data) to avoid generating representations which are biased towards artifacts in unlabeled data. As a result, DDL generates a less biased representation that is beneficial for both reconstruction and prediction loss minimization, thus leading to better predictive phenotyping performance. This is in contrast to existing semi-supervised ML methods [Mairal *et al.*, 2009; Tariyal *et al.*, 2016; Dligach *et al.*, 2015], which perform training with separate supervised and unsupervised steps, and are therefore vulnerable to biased representations caused by artifacts in unlabeled data.

To demonstrate the aforementioned advantages of DDL, we evaluated its performance and compared it against those of a set of selected state-of-the-art predictive phenotyping baselines on an extensive benchmark (Section 3) comprising multiple real-world EHR datasets and a wide variety of predictive phenotyping tasks (e.g., heart failure classification, mortality and sequential prediction). The reported results demonstrate improved prediction performance of DDL consistently over all baselines, which provides strong empirical evidence to support our key contribution statement that unlabeled data can be coupled with labeled data via semi-supervised learning to boost the performance of predictive phenotyping.

2 Method

This section presents the technical details of our developed DDL framework. In particular, our notations are first introduced in Section 2.1. Section 2.2 presents our developed *deep dictionary encoder* that generates *deep dictionary representations* for EHR data. Section 2.3 then presents a *deep dictionary decoder* that maps the generated *deep dictionary representations* back to EHR data, and optimizes it via minimizing the reconstruction loss. Section 2.4 develops a *deep dictionary predictor* that maps each patient's *deep dictionary representation* to relevant target outcomes. Finally, Section 2.5 presents a collaborative training architecture (see Fig. 1) that jointly trains all the aforementioned components, thus allowing them to converge on the best *deep dictionary representation* that minimizes both reconstruction and prediction losses.

2.1 Notations and Definitions

Our EHR data comprises medical records of N different patients. The medical record of each patient $n = 1 \dots N$ is represented as: the input $\mathbf{X}_n \triangleq [\mathbf{x}_n^{(1)}; \mathbf{x}_n^{(2)}; \cdots; \mathbf{x}_n^{(k_n)}]$ denotes a collection of patient's records for his/her past visits to the clinic, and k_n is the number of the n-th patient's visits. Each visit record $\mathbf{x}_n^{(i)} \in \{0,1\}^p$ can be represented as a multi-hot vector that indicates whether the patient was associated with a particular medical code during his/her i-th visit to the clinic. There are p unique medical codes. The corresponding output/label $\mathbf{y}_n \in \{0,1\}^c$ of a patient's EHR data $\mathbf{X}_n$ is also a multi-hot vector that indicates whether the patient was diagnosed with a certain target disease or condition. There are c unique target diseases or conditions. In our setting, the label $\mathbf{y}_n$ is only available for a small subset of $M < N$ patients with $n = 1, \dots, M$, which is designated as our labeled training dataset $\mathcal{D} \triangleq (\mathbf{X}_n, \mathbf{y}_n)_{n=1}^M$. Given the labeled and unlabeled EHR datasets of previous patients, i.e. $\mathcal{D} \triangleq (\mathbf{X}_n, \mathbf{y}_n)_{n=1}^M$ and $\mathcal{U} \triangleq (\mathbf{X}_n,)_{n=1}^N$, respectively, the aim of the predictive phenotyping task is to learn a latent function that maps from a patient's EHR data $\mathbf{X}$ to an output vector $\mathbf{y} \in \{0,1\}^c$ that characterizes his/her phenotype accurately.

2.2 Deep Dictionary Encoder

This section develops a *deep dictionary encoder* that can succinctly captures embedded representations from the patient's *unlabeled* EHR data $\mathcal{U} = (\mathbf{X}_n)_{n=1}^N$ coherently and in accordance with each other. This will help establish a parsimonious set of hidden-layer patient prototypes, which constitutes our *deep dictionary representation*. This can be achieved by combining ideas from both deep and dictionary learning, which is an emerging paradigm of predictive deep models in computer vision.

Intuitively, the key idea is to first exploit the capability of DL model to embed complex data (e.g., EHR data with varying length due to different numbers of clinical visits among different patients) into fixed-size, low-level embeddings. To incorporate high-level representations for better predictive performance, we use dictionary learning to further decompose these embeddings into fundamental patterns that succinctly characterize a space of high-level features. The patient embeddings can then be projected onto this space and the projection coefficients can be leveraged as high-level features to improve the cognitive capability of the predictive model.

To represent each patient, we employ an RNN with Long Short Term Memory (LSTM) architecture [Hochreiter and Schmidhuber, 1997], which is well-known for its ability to capture long-range dependencies within longitudinal data such as patient's EHR. In particular, the LSTM module generates a latent embedding $\mathbf{g}_n$ from the multi-hot encoded patient record $\mathbf{X}_n$,

$$\mathbf{g}_n = \text{LSTM}(\mathbf{X}_n) \;=\; \text{LSTM}\left(\mathbf{x}_n^{(1)}, \mathbf{x}_n^{(2)}, \dots, \mathbf{x}_n^{(k_n)}\right) , \quad (1)$$

where $\mathbf{g}_n \in \mathbb{R}^d$ is the output of the LSTM, which is used as the embedded, fixed-length patient representation. In the above equation, we concatenate the multi-hot vectors (corresponding to the patient's different clinical visits) into an extended input vector for the LSTM. Based on the above embedded representation, we further adopt dictionary learning to extract both a dictionary of hidden-layer prototypes (which is patient-independent) and a collection of weight vectors (one per patient), which specifies how these prototypes can be combined to characterize a particular patient accurately. This is achieved via minimizing the following regularized projection loss,

$$\mathcal{L}_d\left(\mathbf{D}, \{\mathbf{r}_n\}_{n=1}^N\right) \triangleq \sum_{n=1}^{N} \left(\frac{1}{2}\left\|\mathbf{g}_n - \mathbf{D}\mathbf{r}_n\right\|_2^2 + \lambda_1 \|\mathbf{r}_n\|_1\right) + \lambda_2 \left\|\mathbf{D}\right\|_{\text{F}}^2 , \quad (2)$$

with respect to $\{\mathbf{r}_n\}_{n=1}^N$ ($\mathbf{r}_n \in \mathbb{R}^k$) and $\mathbf{D} \in \mathbb{R}^{d \times k}$, which denote the weight vectors (or ***sparse codes***) that characterize

each patient and the dictionary of *patient prototype* represented in the embedded space, respectively. To avoid generating trivial solutions, the above loss is also regularized by penalizing the complexities $\lambda_1\|\mathbf{r}_n\|_1$ and $\lambda_2\|\mathbf{D}\|_{\mathrm{F}}^2$ of the sparse codes and dictionary with λ_1 and λ_2 being the regularization parameters. The embedded patient prototype can then be mapped back to the patient space via a *deep dictionary decoder* (see Section 2.3). Note that the above loss in Eq. 2 is convex in $\mathbf{D}$ given $\{\mathbf{r}_n\}_{n=1}^N$ and vice versa. Thus, Eq. 2 can be optimized (with local convergence guarantee) via alternating minimization [Chatterji and Bartlett, 2017].

2.3 Deep Dictionary Decoder

To map the patient *deep dictionary representation* $(\mathbf{D}, \mathbf{r}_n)$ back to the original patient space, we develop a *deep dictionary decoder* module which specifically maps $(\mathbf{D}, \mathbf{r}_n)$ to a vector $\mathbf{q}_n \triangleq [\mathbf{q}_n^{(1)} \dots \mathbf{q}_n^{(p)}] \in [0,1]^p$ of probability scores such that $\mathbf{q}_n^{(i)} \in [0,1]$ denotes the probability that medical code i contributes actively to the patient's predicted outcome $\mathbf{y}_n$. We parameterize the decoder $\mathbb{D}(\mathbf{D}, \mathbf{r}_n)$ as a neural network with a fully-connected layer followed by a sigmoid activation,

$$\mathbf{q}_n \;\triangleq\; \mathbb{D}(\mathbf{D}, \mathbf{r}_n; \mathbf{W}) \;\triangleq\; \sigma\left(\mathbf{W}\mathbf{D}\mathbf{r}_n\right) , \qquad (3)$$

where the sigmoid operator $\sigma(x) \triangleq 1/(1+\exp(-x))$ is applied point-wise to each element of the logit vector $\mathbf{W}\mathbf{D}\mathbf{r}_n$. The above neural network is parameterized by the weight $\mathbf{W}$ of the dense, fully-connected layer. To optimize $\mathbf{W}$, we generate the augmented dataset $\{(\mathbf{D}, \mathbf{r}_n), \overline{\mathbf{X}}_n\}_{n=1}^N$, which can be generated via the deep dictionary embedding technique presented in Section 2.2. The augmented dataset can then be used to train the above neural network via back-propagation with the following cross-entropy reconstruction loss,

$$\mathcal{L}_r\left(\mathbf{W}, \mathbf{D}, \{\mathbf{r}_n\}_{n=1}^N\right) = \sum_{n=1}^{N}\sum_{i=1}^{p} \overline{\mathbf{X}}_n^{(i)} \log \mathbf{q}_n^{(i)} + \left(1 - \overline{\mathbf{X}}_n^{(i)}\right) \log\left(1 - \mathbf{q}_n^{(i)}\right) , \qquad (4)$$

where $\overline{\mathbf{X}}_n^{(i)} \in [0,1]$ is the average occurrence of medical code i over all clinical visits of patient n. Optimizing Eq. 4 thus yields the *deep dictionary decoder*. To use this *deep dictionary decoder* on an unseen patient, we project the patient embedded representation $\mathbf{g}$ on $\mathbf{D}$ via minimizing $\|\mathbf{g} - \mathbf{D}\mathbf{r}\|_2^2$ with respect to $\mathbf{r}$, which can be solved analytically.

2.4 Deep Dictionary Predictor for Labeled Data

Given the *deep dictionary representation* $(\mathbf{D}, \mathbf{r}_n)$ of labeled training data $\mathcal{D} = (\mathbf{X}_n, \mathbf{y}_n)_{n=1}^M$, we can further build a *deep dictionary predictor* that maps from a patient's sparse code $\mathbf{r}_n$ to a vector $\mathbf{o}_n$ of predicted probabilities that the patient will be associated with each target outcomes (e.g., disease or mortality). This is achieved by parameterizing the predictor with a fully-connected layer followed by a soft-max activation,

$$\mathbf{o}_n \;=\; \mathrm{softmax}\left(\mathbf{V}\mathbf{r}_n + \mathbf{b}\right) \;\in\; \mathbb{R}^c , \qquad (5)$$

where $\mathbf{V} \in \mathbb{R}^{c\times k}$ and $\mathbf{b} \in \mathbb{R}^c$ are the weight matrix and bias vector, respectively. These parameters can then be learned via

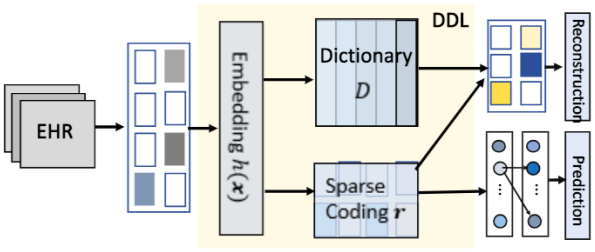


Figure 1: The architecture of our DDL framework includes 3 interconnected modules: (1) an encoder that generates a deep dictionary representation for a patient's EHR data, (2) a decoder and (3) a predictor which are both connected to the encoder, which allows them to collaboratively decide on the best representation that minimizes both reconstruction and minimization losses.

minimizing the cross-entropy between the predicted probabilities $\mathbf{o}_n$ and the ground-truth label $\mathbf{y}_n$,

$$\mathcal{L}_c\left(\mathbf{V}, \mathbf{b}, \{\mathbf{r}_n\}_{n=1}^M\right) \;\triangleq\; -\sum_{n=1}^{M}\sum_{i=1}^{c} \mathbf{y}_n^{(i)} \log \mathbf{o}_n^{(i)} , \qquad (6)$$

with respect to $\mathbf{V}$ and $\mathbf{b}$. c is number of prediction targets, and $n = 1 \dots M$ is the patient index in the *labeled* training data $\mathcal{D}$ with $\mathbf{y}_n \in \{0,1\}^c$ denote the corresponding patient label characterizing his/her phenotype. $\mathbf{y}_n^{(i)}$ and $\mathbf{o}_n^{(i)}$ are the i-th entries of $\mathbf{y}_n$ and $\mathbf{o}_n$, respectively.

2.5 Collaborative Prediction and Reconstruction

This section develops an collaborative architecture that connects the previously developed *deep dictionary decoder* and *deep dictionary predictor* using a common layer of *deep dictionary encoder* (see Fig. 1).

This allows us to optimize the above components simultaneously so that the deep dictionary encoder could interact with both the decoder and predictor modules to figure out a viable communication medium (i.e., the dictionary) that allows them to reach a consensus. Intuitively, the encoder plays the role of a mediator that suggests communication options for the decoder and predictor, and depending on the resulting quality of communication between them (i.e., the incurred prediction and reconstruction losses), the encoder will revise the communication medium until it facilitates an acceptable consensus between the decoder and predictor. This is in contrast to a naive solution that trains these components separately, which does not allow communication/coordination between the encoder, decoder and predictor; and therefore, cannot guarantee that the learned results would be optimized for both reconstruction and prediction.

To facilitate the aforementioned coordination, we aggregate the loss functions of these components (i.e., encoder, decoder and predictor) to generate a combined performance feedback, which can be exploited to update them jointly via stochastic gradient back-propagation,

$$\mathcal{L}_\ell \;=\; \eta_d\mathcal{L}_d \;+\; \eta_r\mathcal{L}_r \;+\; \eta_c\mathcal{L}_c \, , \qquad (7)$$

Algorithm 1 DDL ($T_{\max}$ – max no. of optimizing iterations)

Input: randomized LSTM, $\mathbf{W}$, $\mathbf{D}$, $\mathbf{V}$ and $\mathbf{b}$.
for $t = 1 \dots T_{\max}$ **do**
  sample mini-batch $\mathcal{M} \subseteq \{\mathbf{X}_1, \dots, \mathbf{X}_N\}$.
  for $\mathbf{X}_n \in \mathcal{M}$ **do**
    Encoder: Estimate Embedded Representation
    $\mathbf{g}_n \leftarrow \text{LSTM}(\mathbf{X}_n) = \text{LSTM}\left(\mathbf{x}_n^{(1)}, \mathbf{x}_n^{(2)}, \dots, \mathbf{x}_n^{(k_n)}\right)$
    Encoder: Estimate Sparse Code Representation
    $\mathbf{r}_n \leftarrow \arg\min_{\mathbf{r} \in \mathbb{R}^k} \|\mathbf{g}_n - \mathbf{D}\mathbf{r}\|_2^2 + \lambda_1 \|\mathbf{r}\|_1$ using Alg. 2.
    Decoder: Reconstruction Data
    $\mathbf{q}_n \leftarrow \mathbb{D}(\mathbf{D}, \mathbf{r}_n; \mathbf{W})$ via Eq. 3
    Predictor: Estimate Prediction Probabilities
    if $\mathbf{X}_n \in \mathcal{D}$ (i.e., $\mathbf{X}_n$ is labeled) **then**
      $\mathbf{o}_n \leftarrow \text{softmax}(\mathbf{V}\mathbf{r}_n + \mathbf{b})$ via Eq. 5
    end if
  end for
  Compute $\mathcal{L}$'s stochastic gradient using the above
  Update all parameters via stochastic gradient descent.
end for
Output: optimized LSTM, $\mathbf{W}$, $\mathbf{D}$, $\mathbf{V}$ and $\mathbf{b}$.

where $\mathcal{L}_\ell$ is parameterized by $\mathbf{W}, \mathbf{D}, \mathbf{V}, \mathbf{b}$ and $\{\mathbf{r}_n\}_{n=1}^N$ for which the projection loss $\mathcal{L}_d$ only depends on $(\mathbf{D}, \{\mathbf{r}_n\}_{n=1}^N)$, the reconstruction loss $\mathcal{L}_c$ depends on $(\mathbf{W}, \mathbf{D}, \{\mathbf{r}_n\}_{n=1}^N)$, and the prediction loss $\mathcal{L}_c$ depends on $(\mathbf{V}, \mathbf{b}, \{\mathbf{r}_n\}_{n=1}^M)$. The extra hyper-parameters η_d, η_r and η_c are manually tuned to trade-off between individual losses.

In addition, note that the projection and reconstruction losses do not depend on the training output and can therefore be pre-trained in advance using only unlabeled data $\mathcal{U}$. In this case, the loss function reduces to

$$\mathcal{L}_u \;=\; \eta_d \mathcal{L}_d \;+\; \eta_r \mathcal{L}_r \,, \tag{8}$$

which is first minimized (using unlabeled data) with respect to $\mathbf{D}, \mathbf{W}$ and $\{\mathbf{r}_n\}_{n=1}^N$ to generate a good starting point for these parameters before further optimizing them via Eq. 7, and in accordance with the predictor's parameters $(\mathbf{V}, \mathbf{b})$ using labeled data, thus still allowing end-to-end training that simultaneously optimizes both the supervised and unsupervised components of DDL (albeit with a starting point generated from pre-training its unsupervised component). The above process is, however, not computationally efficient due to the large number of optimizing parameters, which consequently results in a very slow convergence rate if we train all these parameters from scratch. To address this issue, we instead adopt a different approach which first computes a warm-start for the sparse codes $\{\mathbf{r}_n\}_{n=1}^N$ (as detailed below) as a good starting point to initiate gradient back-propagation on the entire network. The main algorithm is shown in Algorithm 1.

To initialize the sparse code $\mathbf{r}_n$ for each patient, we fix the dictionary $\mathbf{D}$ and the data embedding layer of the network (hence, the patient's embedding representation $\mathbf{g}_n$). Thus, given $(\mathbf{D}, \mathbf{g}_n)$, the objective in Eq. 7 is convex with respect to $\mathbf{r}_n$ and reduces the following form,

$$\mathbf{r}_n \;=\; \arg\min_{\mathbf{r} \in \mathbb{R}^k} \left(\frac{1}{2} \|\mathbf{g}_n - \mathbf{D}\mathbf{r}\|_2^2 \;+\; \lambda_1 \|\mathbf{r}\|_1\right), \tag{9}$$

which can be solved efficiently using proximal gradient method [Parikh and Boyd, 2014], as detailed in Algorithm 2 below. The use of proximal gradient descent is a technical necessity in this context since (9) is not differentiable everywhere due to the L_1 regularization $\lambda_1 \|\mathbf{r}\|_1$. Proximal gradient descent sidesteps this issue by decomposing (9) into two parts: (a) $f(\mathbf{r}) \triangleq 1/2 \|\mathbf{g}_n - \mathbf{D}\mathbf{r}\|_2^2$ which is differentiable, and (b) $h(\mathbf{r}) \triangleq \lambda_1 \|\mathbf{r}\|_1$ which is not differentiable. Thus, starting from a random initialization of $\mathbf{r}$, we can use gradient descent on the first part to update $\mathbf{r}$ with respect to the differentiable part $f(\mathbf{r})$ and then update it with respect to the non-differentiable part $\lambda_1 \|\mathbf{r}\|_1$ via the following proximal operator,

$$\mathbf{r} \leftarrow \mathbf{prox}_{\lambda_1 h}(\mathbf{r}) \triangleq \arg\min_{\mathbf{r}'} \left(f(\mathbf{r}') \;+\; \frac{1}{2}\|\mathbf{r}' - \mathbf{r}\|_2^2\right), \tag{10}$$

which can be solved analytically. Interested readers are referred to [Parikh and Boyd, 2014] for further details on the intuition and rationality behind proximal gradient descent.

3 Experiment

In this section, we empirically evaluate the performance of DDL against several state-of-the-art baseline methods on 3 healthcare datasets, Heart Failure (HF) [Ma *et al.*, 2018], MIMIC-III and a subset of Truven MarketScan Data[1], which contain 16794, 58000 and 72179 EHR samples, respectively. The numbers of clinical variables (p) in HF, MIMIC-III and TRUVEN are 1865, 283 and 283, respectively. In HF and MIMIC-III, the task is to predict the HF and mortality binary outcomes. For TRUVEN data, the prediction task is to predict which clinical variables are positive in the patient's latest visit given data of his/her past visits.

3.1 Experiment Settings

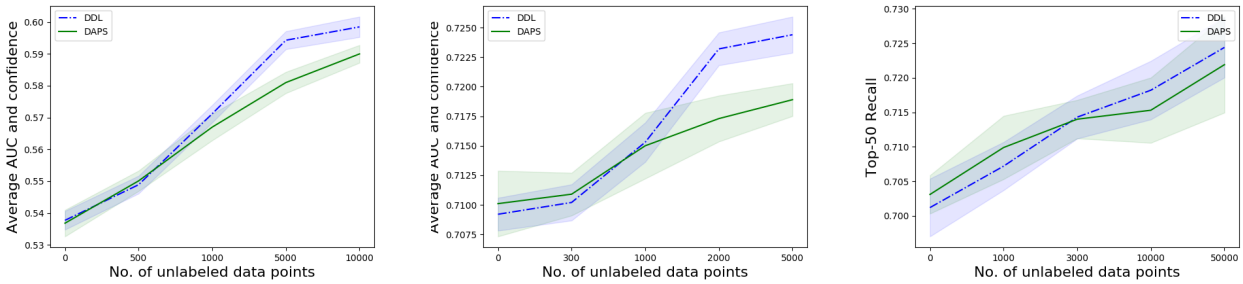
For each experiment, we randomly generate 5 different partitions of the entire dataset into training, validation and testing sets with a $7:1:2$ ratio. The reported result of each tested method is its averaged result over 5 independent runs corresponding to the 5 different data partitions. Our method is implemented by Tensorflow 1.9.0 and Python 3.5[2]; and tested

[1] https://marketscan.truvenhealth.com/

[2] Code is available at https://github.com/futianfan/dictionary.

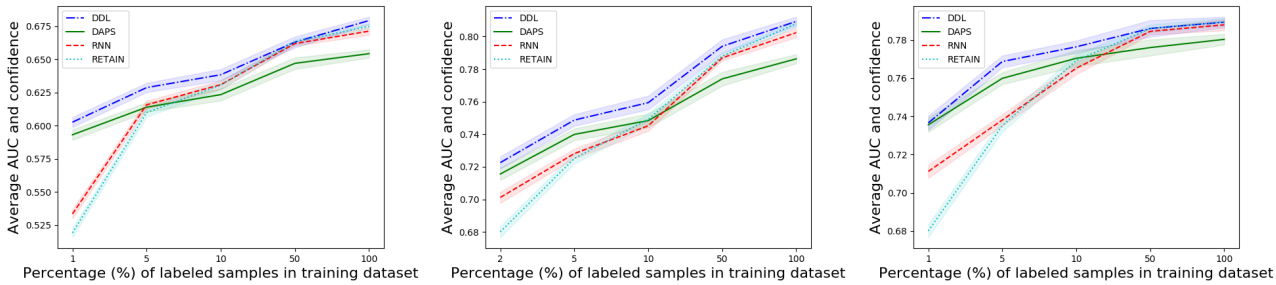
Algorithm 2 Proximal Optimization ($\mathbf{r}$, $T_{\max}$)

Initialize learning rates $\xi^1, \dots, \xi^{T_{\max}}$
for $k = 1 \dots T_{\max}$ **do**
  $\mathbf{r} \;=\; \mathbf{r} \;-\; \xi^k \nabla_{\mathbf{r}} f(\mathbf{r}) \;=\; \mathbf{r} \;-\; \xi^k \mathbf{D}^\top (\mathbf{g} - \mathbf{D}\mathbf{r})$
  $\mathbf{r} \;=\; \text{prox}_{\xi^k \lambda_1 h}(\mathbf{r})$
  if converge **then**
    break
  end if
end for



(a) (b) (c)

Figure 2: Graphs of average accuracy with 95% confidence interval of DDL and DAPS on (a) HF, (b) MIMIC-III, and (c) TRUVEN datasets. The reported performance is generated using 150 labeled data samples and varying number of unlabeled data samples.



(a) (b) (c)

Figure 3: Graphs of average accuracy (with 95% confidence interval) of DDL and baseline methods on (a) HF, (b) MIMIC-III, and (c) TRUVEN datasets. In all experiments, the size of training dataset is fixed while the fraction of labeled data is being varied.

on an Intel Xeon E5-2690 machine with 256G RAM and 8 NVIDIA Pascal Titan X GPUs. We evaluate each method using its best fine-tuned hyper-parameter configurations. The best configurations of DDL are described below.

For **HF** dataset, the no. of hidden units of DDL's RNN component is set to 100. Its dictionary size is set to 10. Its learning rate for gradient back-propagation on the aggregate loss (Eq. 7) is set to be $1e-2$. To trade-off between projection, reconstruction and prediction losses, we set $\eta_d = 1e-1$, $\eta_c = 1$ and $\eta_r = 1e-3$ in Eq. 7. For the projection loss $\mathcal{L}_d$ in Eq. 2, the regularization hyper-parameters are set as $\lambda_1 = 5e-2$ and $\lambda_2 = 1e-3$. For **MIMIC-III** dataset, we use the same configuration but with the following minor changes on learning rate ($5e-2$) and trade-off coefficients ($\eta_d = \eta_c = 1$ and $\eta_r = 2e-3$) between individual losses of DDL. For **TRUVEN** dataset, we also use the similar configuration but with the dictionary and RNN component's hidden sizes set to be 15 and 200, respectively. In addition, the trade-off coefficients in Eq. 7 are also adjusted to $\eta_d = 1e-1$ and $\eta_c = \eta_r = 1$. The batch sizes of DDL's stochastic gradient descent on **HF** and **MIMIC-III** are both set to be 32, while on **TRUVEN**, it is set to be 64 (since **TRUVEN** dataset is larger than the others).

3.2 Baseline Methods

To demonstrate the advantage of DDL, we evaluate and compare its performance with those of the baseline methods, which includes (a) state-of-the-art deep phenotyping methods such as Doctor-AI (**RNN**) [Choi *et al.*, 2016a], Reverse Time Attention RNN (**RETAIN**) [Choi *et al.*, 2016b], Denoising Autoencoders for Phenotype Stratification (**DAPS**) [Beaulieu-Jones *et al.*, 2016], and (b) two traditional ML methods featuring Logistic Regression (**LR**) with L_2 regularizer and Dictionary Learning (**DictLearn**) [Mairal *et al.*, 2009]. The **LR**, **DictLearn** and **DAPS** baselines were trained using the aggregated feature vector $\tilde{\mathbf{X}}_n \in \{0,1\}^p$ (instead of using the original input $\mathbf{X}_n$), which was derived from the original EHR data $\mathbf{X}_n$ such that $\tilde{\mathbf{X}}_n^{(i)} = 1$ if the i-th clinical variable occurred at least once in $\mathbf{X}_n$'s multiple clinical visit. Otherwise, $\tilde{\mathbf{X}}_n^{(i)} = 0$. Furthermore, to showcase the effectiveness of training DDL's unsupervised and supervised components together (instead of separately), we also include in the set of baselines a simplified version of DDL that excludes the decoder module. In this case, the aggregated loss function reduces to $\mathcal{L}_\ell = \eta_d \mathcal{L}_d + \eta_c \mathcal{L}_c$.

3.3 Comparison in Supervised Setting

This section evaluates and reports the performance of DDL and baseline methods in supervised setting. That is, the training only uses EHR data with label. For experiments on **HF** and **MIMIC-III** datasets, the performance of each method is measured by the standard ROC-AUC (Area Under the Receiver Operating Characteristic Curve) score for binary classification where higher score implies better performance. For **TRUVEN**

Model	HF	MIMIC-III
LR	0.636 ± 0.003	0.763 ± 0.004
DictLearn	0.634 ± 0.004	0.771 ± 0.006
DAPS	0.651 ± 0.002	0.794 ± 0.004
RNN	0.668 ± 0.004	0.807 ± 0.004
RETAIN	0.675 ± 0.003	0.817 ± 0.004
DDL	$\mathbf{0.682 \pm 0.004}$	$\mathbf{0.819 \pm 0.004}$
DDL w.o. decoder	0.669 ± 0.006	0.813 ± 0.004

Table 1: Averaged prediction accuracy (ROC-AUC) of DDL and baseline methods (with standard deviation) evaluated on **HF** and **MIMIC-III** datasets. Higher ROC-AUC implies better performance.

Model	HF	MIMIC-III
LR	34.3 ± 2.1	21.2 ± 1.9
DictLearn	49.3 ± 2.9	31.2 ± 3.0
DAPS	125.0 ± 5.6	86.4 ± 2.3
RNN	64.6 ± 8.4	25.6 ± 3.2
RETAIN	100.7 ± 10.1	79.4 ± 14.3
DDL	234.4 ± 21.3	178.3 ± 11.0
DDL w.o. decoder	183 ± 14.1	128.3 ± 6.0

Table 2: Averaged training time (sec) incurred by DDL and baseline methods (with standard deviation) on **HF** and **MIMIC-III** datasets.

datasets, the final prediction is generated by combining the results of individual binary classification tasks (one for each variable in the patient's last clinical visit). In particular, the top k variables with largest predicted probabilities to be positive are treated as positive labels while the others are associated with negative labels in our multi-label prediction. Thus, the performance of each method on **TRUVEN** dataset is measured using the following top-k recall metric:

$$\text{top-k recall} \triangleq \frac{\#\text{ of true positives in top k prediction}}{\#\text{ of true positives}} \quad (11)$$

The averaged performance (with standard deviation) over 5 different runs of all tested methods are reported in Table 1 for **HF** and **MIMIC-III** datasets, and in Table 3 for **TRUVEN** dataset. The incurred training time for all methods are reported in Table 2. The reported results shown that: (1) DDL consistently outperforms all baseline methods in terms of prediction accuracy (ROC-AUC) on all datasets, which demonstrates the advantage of using a hybrid deep dictionary learning in predictive phenotyping; (2) among the baselines, DL-based methods perform significantly better than traditional ML baselines such as **LR** and **DictLearn**, which justifies the choice of using DL as the base model (in our framework) to be combined with dictionary learning; and (3) DDL with decoder achieved better results than DDL without decoder, thus supporting our statement earlier that jointly training both supervised and unsupervised components (instead of training them separately) allow them to coordinate on a better data representation.

3.4 Comparison in Semi-Supervised Setting

To showcase the advantage of leveraging unlabeled data, we design two scenarios to demonstrate it empirically. In the first scenario, we fix the number of labeled data samples and observe the changes in DDL's and baseline methods' performance when the number of unlabeled data samples varies. The

Model	Top-50 recall	Top-30 recall	Time
LR	0.753 ± 0.002	0.605 ± 0.002	43.0 ± 1.2
DictLearn	0.752 ± 0.002	0.610 ± 0.002	84.2 ± 2.0
DAPS	0.773 ± 0.003	0.621 ± 0.002	232 ± 34
RNN	0.783 ± 0.003	0.630 ± 0.002	113.0 ± 8.2
RETAIN	0.786 ± 0.007	0.632 ± 0.005	134.4 ± 12.0
DDL	$\mathbf{0.791} \pm 0.004$	$\mathbf{0.634} \pm 0.003$	624 ± 34
DDL w.o. decoder	0.780 ± 0.05	0.627 ± 0.005	427 ± 31

Table 3: Averaged prediction accuracy (top-k recall) and incurred time (sec) of DDL and baseline methods (with standard deviation) on **TRUVEN** dataset. Higher top-k recall implies better performance.

results are plotted in Figure 2, which show the tested methods' average prediction accuracies of 3 independent runs and their confidence intervals. It can be observed that across all datasets, the prediction performance increases as more unlabeled data is used for training, thus supporting our claim earlier that unlabeled data can be exploited as an extra source of information to improve performance of predictive phenotyping models.

In the second scenario, we fix the total number of training data samples while varying the percentage of labeled samples within the training set, and observe how the tested methods' performance changes accordingly. From the results in Figure 3 above, it shows that DDL outperforms other methods most significantly when the amount of labeled data is limited: when this happens, the performance of **RNN** and **RETAIN** degrade the most since they cannot make use of unlabeled data. **DAPS** on the other hand can leverage unlabeled data and performs better than **RNN** and **RETAIN** but still worse than DDL since it does not train both supervised and unsupervised components, thus being vulnerable to artifacts in unlabeled data.

4 Related Works

In this section, we briefly discuss the success use of deep learning in predictive phenotyping. Choi *et al.* [2016a] used Recurrent Neural Network (RNN) to modelling temporal data in Electronic Health Record (EHR). Choi *et al.* [2016b]; Ma *et al.* [2017]; Choi *et al.* [2017] uses attention mechanisms to detect important visits and clinical variables. Choi *et al.* [2018] utilized multilevel structures to modelling EHR data in a finer-grained manner. Ma *et al.* [2018] extracts comprehensive patient pattern via using time-aware neural architectures. Most existing works only pay attention to labeled data. In this paper, we attempt to improve the predictive phenotyping in an orthogonal direction focused on leveraging unlabeled data.

5 Conclusion

This paper addressed the label-insufficiency issue in predictive healthcare where unlabeled data are abundant but labeled data is limited. To combine both labeled and unlabeled data for better predictive performance, we propose a deep dictionary learning (DDL) framework that utilizes unlabeled data to learn more generalizable representation of data for the predictive model. For example DDL was evaluated on real-world healthcare datasets (MIMIC-III, Heart Failure and Truven) for disease prediction. The results consistently show that the representations learned from unlabeled data generalize better and improve the predictive accuracy.

Acknowledgements

This work was supported by the National Science Foundation award IIS-1418511, CCF-1533768 and IIS-1838042, the National Institute of Health award 1R01MD011682-01 and R56HL138415.

References

Inci M Baytas, Cao Xiao, Xi Zhang, Fei Wang, Anil K Jain, and Jiayu Zhou. Patient subtyping via time-aware lstm networks. In *proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 65–74, September 2017.

Brett K Beaulieu-Jones, Casey S Greene, et al. Semi-supervised learning of the electronic health record for phenotype stratification. *Journal of biomedical informatics*, 64:168–178, October 2016.

Niladri Chatterji and Peter L Bartlett. Alternating minimization for dictionary learning with random initialization. In *Advances in Neural Information Processing Systems*, pages 1997–2006, December 2017.

Edward Choi, Mohammad Taha Bahadori, Andy Schuetz, Walter F Stewart, and Jimeng Sun. Doctor ai: Predicting clinical events via recurrent neural networks. In *Machine Learning for Healthcare Conference*, pages 301–318, May 2016.

Edward Choi, Mohammad Taha Bahadori, Jimeng Sun, Joshua Kulas, Andy Schuetz, and Walter Stewart. Retain: An interpretable predictive model for healthcare using reverse time attention mechanism. In *Advances in Neural Information Processing Systems*, pages 1493–1501, December 2016.

Edward Choi, Mohammad Taha Bahadori, Le Song, Walter F Stewart, and Jimeng Sun. Gram: Graph-based attention model for healthcare representation learning. In *proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 787–795, September 2017.

Edward Choi, Cao Xiao, Walter Stewart, and Jimeng Sun. Mime: Multilevel medical embedding of electronic health records for predictive healthcare. In *Advances in Neural Information Processing Systems*, pages 4547–4557, December 2018.

Dmitriy Dligach, Timothy Miller, and Guergana K Savova. Semi-supervised learning for phenotyping tasks. In *AMIA Annual Symposium Proceedings*, volume 2015, page 502. American Medical Informatics Association, April 2015.

Cristóbal Esteban, Oliver Staeck, Stephan Baier, Yinchong Yang, and Volker Tresp. Predicting clinical events by combining static and dynamic information using recurrent neural networks. In *2016 IEEE International Conference on Healthcare Informatics (ICHI)*, page 93–101, October 2016.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, April–June 1997.

Zachary C Lipton, David C Kale, Charles Elkan, and Randall Wetzel. Learning to diagnose with lstm recurrent neural networks. *arXiv preprint arXiv:1511.03677*, November 2015.

Fenglong Ma, Radha Chitta, Jing Zhou, Quanzeng You, Tong Sun, and Jing Gao. Dipole: Diagnosis prediction in healthcare via attention-based bidirectional recurrent neural networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1903–1911. ACM, September 2017.

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

Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online dictionary learning for sparse coding. In *Proceedings of the 26th annual international conference on machine learning*, pages 689–696. ACM, June 2009.

Neal Parikh and Stephen Boyd. Proximal algorithms. *Foundations and Trends® in Optimization*, 1(3):127–239, January 2014.

Snigdha Tariyal, Angshul Majumdar, Richa Singh, and Mayank Vatsa. Deep dictionary learning. *IEEE Access*, 4:10096–10109, April–June 2016.