Towards Risk Prediction via 3D Convolutional Neural Networks for Electronic Health Records

Submitted for Blind Review

***Abstract*—The problem of extracting useful clinical representations from longitudinal EHR data, sometimes called computational phenotyping, is an important but challenging task in the healthcare academia and industry. Recent progress in the design and applications of deep learning methods has shown promising results towards solving this problem. In this paper, we investigate the usage of the convolutional neural network (CNN) in modeling longitudinal EHR data. Particularly, we propose a 3D CNN structure, which is featured by spatial pyramid pooling (SPP). Compared with methods of 2D CNNs, the proposed method can effectively and efficiently capture the complicated internal relations in EHRs. Besides, in previous work, the issue of variety in patient record length is addressed by padding zero all vectors to a fixed length. It is well handled with the spatial pyramid pooling, which divides the records into several length sections for respective pooling processing. On cohorts of congestive heart failure and diabetes patients, the experiments produce promising results, demonstrating the effectiveness of both the 3D CNN architecture and the spatial pyramid pooling in risk prediction, compared with several strong baselines.**

I. Introduction

Global health care systems are rapidly adopting Electronic Health Records (EHR), which are systematic collections of longitudinal patient clinical events (e.g., diagnosis, medication, lab tests, procedures, etc.) generated by one or more encounters in any care delivery setting. This has dramatically increased the quantity of clinical data that are available electronically. As a result, *Data-Driven Healthcare (DDH)* [1], i.e., the practice of leveraging big medical data to provide effective personalized care, has been under rapid development and attracted many researchers and institutions utilizing state-of-the-art machine learning and statistical models on a broad set of clinical tasks which are difficult or even impossible to solve with traditional methods [2], [3], [4].

While existing achievements on current data-driven models for healthcare are encouraging, the peculiar properties of EHRs, such as heterogeneity, longitudinal irregularity, inherent noises, and incomplete nature, make it extremely difficult to apply most existing mature models to healthcare compared with other well-developed domains with clean data. To solve those challenges, research and industry communities begin to propose several solutions [5], [6], [7]. Among these efforts, deep learning models tends to be the most exciting and promising solution to those difficult and important tasks. Properly-designed deep neural networks have the prospect of handling these issues if equipped with massive data,

Previous deep learning frameworks, such as phenotype learning [8] and representation learning [9], formed a multi-layer perception (MLP) architecture and applied it to EHRs. Considering the natural temporarily in EHR data, Lipton *et al.* [10] used a Recurrent Neural Network (RNN) for sequence prediction with regular times series of real-valued variables collected from intensive care unit patients. Choi *et al.* [11] proposed an interpretable RNN model to predict the diagnosis code based on medication, lab test and disease information. Soumya *et al.* [12] designed a deep state space model for phenotype learning and handled the data irregularity issue. Several other works [13], [14] also exploited convolutional neural networks, which can capture local temporal dependency of data, to learn the representation for risk prediction or other related tasks.

In this paper, we continue taking the advantage of CNNs to catch local temporal dependency. To further explore the complicated structures in electronic health records, both sequential and hierarchical, we perceive that the correlation of clinical events generally weakens with their distance increasing, which makes the local correlation the most reliable and effective clue for us. Thus we transform 2D CNN to 3D CNN and introduce the new dimension: original EHRs are split into phrases of k events (k is an integer chosen from $1, 2, 3, 4$), which means now a concatenation of *k-event-event* pattern rather than a sequence of events are sent into the network. It turns out that 3D CNN can capture local dependencies more effectively than ordinary CNNs due to 3D architecture which preserves important information contained in event-event patterns, as 2D CNNs can not preclude event-level features from being destroyed more or less in pooling operations. Besides, we consider a mechanism to take the place of padding all EHRs zero to a maximum length, which might backfire, since short EHRs suffer from severely adverse impacts brought out by the padding operation. We utilize Spatial Pyramid Pooling (SPP) [15] to solve the problem, the drastic difference of length of EHRs. Spatial Pyramid Pooling enables our CNN to take in variable-size input, weakening the boundary effect brought by great length variability (the boundary effect is a tough problem for CNNs and might be considered one of the major weaknesses compared with LSTMs).

We evaluate our model on two clinical real-world benchmarks. As we display, our model outperforms other state-of-the-art baselines on both datasets, indicating the effectiveness of the proposed method. The contributions of this paper are summarized as follows: firstly we propose a novel 3D CNN architecture for onset prediction, which explores a new direction of modeling temporal structure on EHRs. The model is extensible and can be utilized on other tasks with similarly structural data. Secondly, to the best of our knowledge, this is

the first study using 3D CNN and SPP in such a task. By processing specific input through according pooling layers, redundancy of information is precluded while elementary features remain untouched.

II. Related works

In this section, we briefly introduce the existing works close to ours in the field of medical prediction and diagnosis, as well as the existing technologies that our method referred.

For diagnosis and prediction of diseases, the most important part is to extract medical phenotyping from patient EHRs. Hripcsak et al. showed in their paper [16] that EHRs made an unprecedented amount of clinical information available for research, and it was pivotal for electronic phenotyping to use it efficiently. An automated prediction model was proposed by Amarasingham et al., which was a classic data-driven model that made great performance on the prediction of 30-day readmission and death for hospitalized heart failure patients [17]. The most accurate and tractable method to extract electronic phenotyping was representing each patient's detection indicators and physical condition with tensor factorization like [18]. Moreover, sequence based representation [19] and temporal matrix based representation [20] were also widely applied. In our work, we combine the advantages of sequence and temporal representation and use word2vec to learn medical feature embeddings. Word2vec is an efficient model designed to characterize words by real vectors, which also can be regarded as coordinates in a multidimensional space. In this vector space, the words with similar contexts are closer to each other and the words with irrelevant meanings have further distance. Since our EHRs data is also sequential data with thousands of different medical events, including medical examinations, lab tests, diagnosis, medications, etc., its data structure is very similar to the structure of text content so we can apply wor2vec model to the large corpus of EHRs data to get the vector representation of each medical feature.

When the medical phenotyping from patient EHRs was obtained, we need an efficient method to make feature extraction and data classification. Deep learning has great superiority in the field of local feature abstraction and temporal data processing. Since the rejuvenation of neural network [21], deep learning has achieved significant progress in many areas [22], such as computer vision [23] and speech r-3D CNN, which is used to analyze each patient's physical condition at different times for early diagnosis of diseases. In our work, we make use of 3D convolutional neural network as well as Spatial Pyramid Pooling (SPP) to process patient's records and extract useful characteristics from them.

3D convolutional neural network, originally invented for action recognition [24], is good at extracting distinguishing characteristics from both spatial and temporal dimensions. Different from traditional 2D CNN, where convolution operations are only applied over one single frame, 3D CNN can generate multi-channel information from continuous video frames. Thus 3D CNN can make use of characteristics captured from both spatial and temporal dimensions to make better prediction. As far as we know, the 3D CNN models has been successfully applied on sentence classification and got good performance. In the sentence classification task, all the sentences are split into a concatenation of phrases before sent to the 3D CNN network. The result showed that 3D CNN model got the best performance compared with other relevant models in the sentence classification work. In our risk prediction task on EHRs data with diagnoses and medications, the characteristics within each medical event feature, together with correlations between features in temporal structures provide us with abundant useful information simultaneously. Additionally, the data structure of EHRs is similar to dataset for sentence classification work, showing that 3D CNN has the ability to be applied on the EHR based risk prediction scenario.

The Spatial Pyramid Pooling (SPP) in our model is based on but different from the SPP in traditional 2D models. In traditional models, Spatial Pyramid Pooling is successfully used to tackle the variable-size input tasks [15]. In addition, by extracting features in multi-angles from one feature map, the SPP can effectively improve the accuracy of object recognition. While in traditional CNN, the input images of arbitrary size are preprocessed by cropping or warping zero to a fixed size. Since the cropping operation may discard useful information and warping may cause severe distortion, we make use of SPP to avoid the troubles above and to process input records with arbitrary size. And in the mean time, we adopt a new method of preprocessing by dividing all the patients medical records into several sections according to statistical results. Then we pad all the records with zeros to the upper boundary in each section separately. Compared with the traditional way in 2D models, we solved the problem of length variety largely. Accordingly, when building the model, we precisely attune the parameters in each layer to process the data coming from different sections in order to guarantee a fixed-size output. Besides, we construct a pyramid structure at the end of the last pooling layer to extract characteristics from multi-angles and multi-scales, making best use of the information extracted by our network.

III. Methodology

This section introduces the main methods we used, including the input and output of our whole system, data preprocessing, and the details for our 3D CNN-SPP network.

A. Input & Output

The raw input to our system is EHR data of each patient. The EHR data is composed of different medical events arranged in chronological order. The length of EHR data vary from patient to patient, and each medical event is represented by a ordinal number like a word index in a vocabulary. The output of our system is the number 1 or 0, representing the patient has the targeted disease or not. In our work, we only apply our model to two important diseases, diabetes and congestive heart failure, as two binary classification tasks separately.

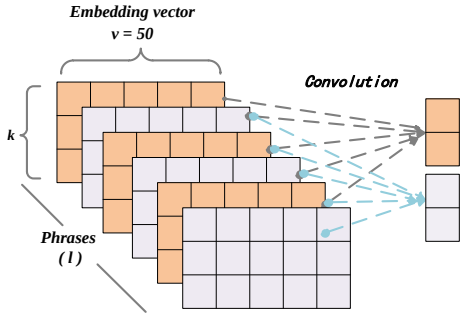


Fig. 1. This picture shows the convolution operation in the first 3D convolution layer. The size of filter in 3D convolution layer is $5 \times 1 \times 2 \times 50$. A window of the same size slides along the phrase sequence. As the picture shows, the convolution applied within each phrase makes the $k \times D$ matrix become a 2 length vector, and the convolution operation applied over the phrase sequence covers 5 phrases each time.

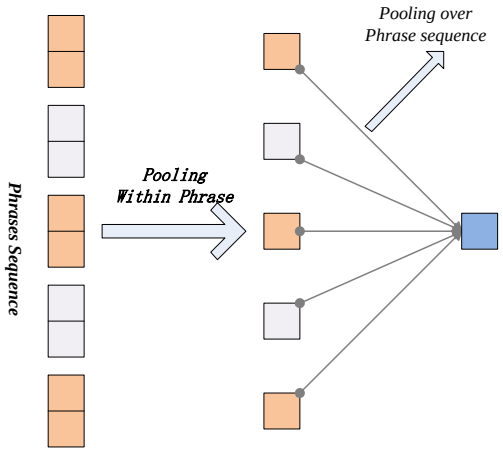


Fig. 2. This picture shows the pooling operation in the first 3D pooling layer. For different length of input, the pooling rate is not the same. As the picture shows, the pooling operation inside the phrase extracts the most prominent feature within one phrase, and the pooling operation applied over the phrase sequence covers several phrases each time to capture the most representative feature among temporal dimension.

B. Data Preprocessing

First, we put the EHRs data of all patients into word2vec to build the embeddings. In our work, the embedding vector size is set to 50 and every medical feature get its precise vector representation. After this operation, the medical records of patients are represented by a list composed of many 50-length vectors. The data in this format is more conducive and convenient to be processed by the neural networks.

Then, we split the 2D medical records list of each patient into phrases of k (chosen from 1, 2, 3, 4) embedding vectors to fit into the 3D CNN model. Now a sequence of k-vectors phrases (3D) rather than a list of embedding vectors (2D) will be put into our network. On this condition, the 3D kernels could apply convolution operations over embedding vectors within each phrase and several adjacent phrases simultaneously. In this way, we can capture reliable characteristics from in-phrase level and across-phrase level in every operation. These captured characteristics will be integrated to form a complete characterization in our 3D CNN-SPP network.

C. The structure of our network

In this section we focus on the data processing procedure in different layers of our network and then display the entire framework in detail.

*1) **3D CNN layer**:* As mentioned above, the input to our CNN model is the record of patient p, which is represented as a 3D temporal and embedding matrix $X_p \epsilon \boldsymbol{R}^{l \times k \times D}$. l is the number of phrases, usually different among the patients (the relationship between medical event numbers, T, and phrase numbers, l, is $l = T - k + 1$). k represents the number of embedding vectors within one phrase, and D is the length of each embedding vector (D=*50*). Each row $x_{ij} \epsilon \boldsymbol{R}^D$, $i \epsilon (1 \dots l)$, $j \epsilon (1 \dots k)$ in X_p is the j*th* embedding vector in the i*th* phrase of that patient's record. The details of input structure are shown in the section IV-A2.

Since embedding dimension contains no spatial or temporal relationship, we only apply convolution operation over the temporal dimension. The first 3D convolution layer has N filters and each filter is defined by $w \epsilon \boldsymbol{R}^{n \times 1 \times h \times v}$, where v equals to the embedding vector's length, showing that output matrix's lowest dimension is a number rather than a vector. The h means that convolution operation is applied over h events inside one phrase. The 1 means that there is no multi-channel information among input data. In the highest dimension, n phrases are covered in every slide of the filter. More details about convolution operation are illustrated in Fig. 1.

For each patient, we will have a output matrix of size:

$$X_p \epsilon \boldsymbol{R}^{(l-n+1) \times N \times (k-h+1) \times 1}, \tag{1}$$

Notice that N filters will result in N channels in the output matrix, and the 3D kernels in the second convolution layer will deal with multi-channel information.

*2) **3D Max-pooling layer**:* Our 3D pooling operation is dissimilar but based on SPP [15]. After convolution layer, we apply max pooling process over phrase sequence and embedding vectors within each phrase simultaneously. More details of max pooling operation are shown in Fig. 2.

The max pooling operation brings us two obvious advantages. To begin with, we can keep the most important features across the time, which is vital for us to handle further extraction works. Moreover, the precisely attuned pooling rates in each layer assure that input with different size lead to the same-sized output.

Since the records of all patients have been segregated into several length sections, our network mainly focuses on applying different pooling rate according to different input length. In our work, when k is equal to 3, there are four length sections, [98,142,186,246]. Let us use l to represent the input with length ($l \epsilon [98, 142, 186, 246]$), and m to show

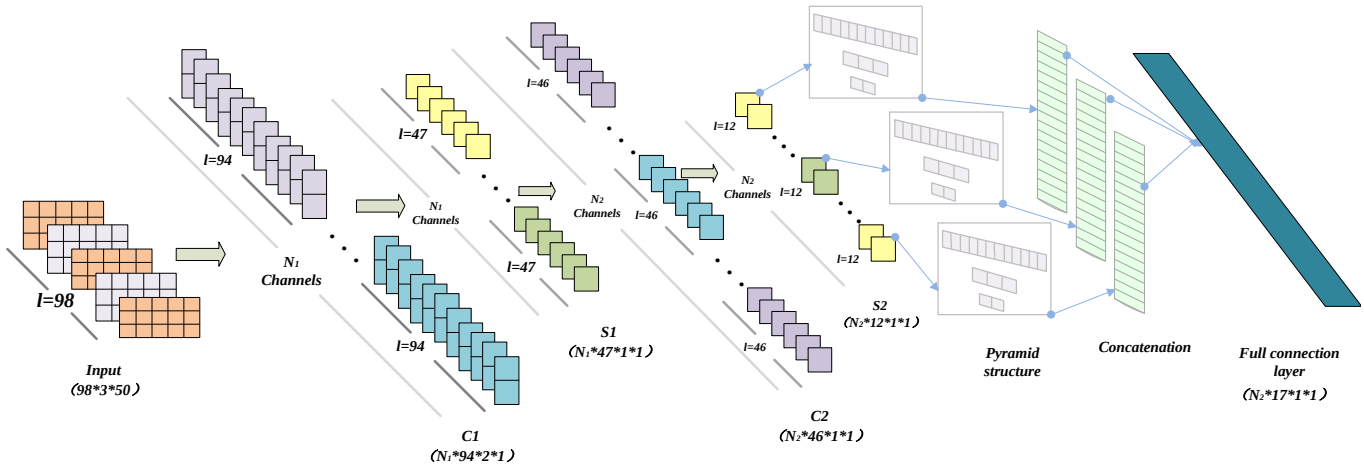


Fig. 3. Our network contains two 3D convolution layers (*C1*, *C2*), two 3D max pooling layers (*S1*, *S2*), pyramid structures and a full connection layer (*F1*). Here we take the example of data procession steps for (*l=98*) section to show our network's details in each layer. The data structure of each layer's output has been shown in brackets. The *C1* layer contains N_1 filters of size *5*1*2*50*, and the *S1* layer's pooling rate is 2 over phrase sequence level, 2 over in-phrase level. The *C2* layer contains N_2 filters of size *2*N_1*1*1*, and the *S2* layer's pooling rate is 4 over phrase sequence level, 1 over in-phrase level. After *S2* layer, Pyramid Structure is built for each channel, and all of them is flattened and concatenated. In the end, the result of concatenation operation with size *N_2*17*1*1* is sent to the full connection layer.

the output length. The following formula briefly illustrates the relationship between pooling rates and input, output.

$$[\frac{l}{m}] = (p_1^l \times p_2^l \times \dots p_i^l), \tag{2}$$

p_i^l could be regarded as the pooling rate in *ith* layer for input length *l*. $[\frac{l}{m}]$ is the operation of rounding outcome up to an integer. In this way, our network can guarantee a fix-sized output by applying different pooling rates accordingly in each layer.

3) ***Framework****:* In this section we display the framework of our 3D CNN-SPP architecture in detail. It's composed of two 3D phrase-level convolution layers (*C1* and *C2*), two 3D phrase-level pooling layers (*S1* and *S2*), and a full connection layer (*F1*) at the end of the network. The details of our network are shown in Fig. 3.

As mentioned above, the input to our network has been pre-trained with word2vec, and split into phrases. When *k* is equal to 3, the sequential length of input, representing the number of phrases, is choosing from [98,142,186,246]. To be specific, firstly, the input with size $l \times 1 \times k \times 50$ enters *C1*, where 1 shows that the number of channels is 1 among input structure, which will be changed according to the filters' number. The *C1* has N_1 filters of size $5 \times 1 \times 2 \times 50$, thus the output size of *C1* layer is $(l-4) \times N_1 \times (k-1) \times 1$ (obviously the output matrix has N_1 channels due to C1's filters). When the data enters the first pooling layer, since the pooling layer will apply different pooling rate according to data's length, we tentatively use $l' \times N_1 \times k' \times 1$ to denote the output of *S1* layer (notice that l' and k' are the division results of *S1* input). Subsequently, the output of *S1* enters the *C2* layer, whose kernel has N_2 filters with size $2 \times N_1 \times 1 \times 1$. Based on the same mechanism, the output size of *C2* is $(l'-1) \times N_2 \times k' \times 1$. In the *S2* layer, according to our pre-determined, the output size of *S2* is $12 \times N_2 \times 1 \times 1$. It is clear that we can get the fixed-size output no matter what the size of input is. Between the *S2* and the final full connection layer (*F1*), we build pyramid structure in every channel to extract the features from multi-angles. To form the pyramid model, we apply a 6-max pooling and a 4-max pooling operation on the $S2$'s output respectively (only aimed at the sequential dimension). Thus, we obtain three feature maps: the 12, 3, 2 in the sequential dimension, and all of them are concatenated and flattened to be sent into full connection layer. Finally in the full connection layer, the data will be reshaped to a one dimension vector with the size $N_2 \times 17 \times 1 \times 1$, and then be processed to get the final prediction result.

The architecture described above is the best one of all the structures we tested during the experiment. In section IV we will show all the contrast tests we did.

IV. Experiments

In this section, we conduct a series of experiments with 3D CNN-SPP on several datasets which have been processed through word2vec. The results demonstrate the effectiveness of our model when compared with several strong baselines. Meanwhile, we also perform experiments on different hold-off period data to evaluate the performance of early prediction. In addition, we investigate more variants of our model for better performance.

A. Experimental Setup

This section mainly focus on the size of our dataset and how we organize them in our work.

*1) **The acquisition of our dataset**:* We apply our model to two important diagnoses, diabetes and congestive heart failure, as two binary classification tasks separately. Firstly, we take EHRs data for 2660 and 5320 patients in the case and control group for heart failure, 2380 and 4760 patients in the case and control group for diabetes. The number of patients in the case and control group are kept at 1:2 to make sure good control effect. Next, we generate each patient's record by making all medical events of one patient into a sequential feature-id list. In order to guarantee the prediction performance, the number of medical events in each patient's record is controlled between 50 and 250. Then we use word2vec to extend each medical event's ordinal number to the corresponding embedding vector. In the end, we obtain a two-dimension *(temporal dimension, embedding dimension)* matrix for each patient's record.

*2) **The structure of input data**:* As we mentioned in the section III-B, we split the two dimensional (temporal and embedding) matrix into phrases of k medical events, thus the original record of each patient is reshaped to three dimension $l \times k \times 50$, where l is chosen from $[100-k+1, 144-k+1, 188-k+1, 248-k+1]$. Finally, the size of dataset sent to our model could be represented as $N \times l \times 1 \times k \times 50$, N is the number of patients, the third dimension represents the number of channels of the data, which will be changed by the number of filters in 3D convolution layer.

Before starting our training, we split the dataset into training, validation, and test subsets by 7:1:2, believing this empirical choice will lead to a satisfying result.

B. Implementation details

*1) **Over-fitting Controlling**:* In our work, we mainly use three methods to control the over-fitting problems.

* **Dropout**: We use dropout layer to randomly hide some parameters in each layer at every batch during training, and the hidden parameters will be kept and updated in later batches training [25]. In this way, all the parameters will not be updated at the same time, reducing the occurrence of over-fitting.
* **L2 regression**: We add one regularization term at the end of cost function. This regularization term could be regarded as a penalty function, which will result in smoother weights. Regularized networks are constrained to build relatively simple models based on patterns seen often in the training data, and are resistant to learning peculiarities of the noise in the training data.
* **Early stop**: We test the prediction accuracy at a fixed frequency during the training process. If we find current prediction performance is not as good as former performance in a continuous period of time, we will terminate the training process immediately to avoid over-fitting.

*2) **Activation Function**:* In our work, we selected rectified linear units (ReLU) as our activation function. Compared with the traditional hyperbolic tangent function (tanh), ReLU will give sparseness to the network, in the mean time, reduce the interdependence between parameters. As a result, ReLU will alleviate the over-fitting problem and make the network easier to be optimized.

*3) **Evaluating the model**:* In our work, we take heart failure and diabetes as examples to evaluate the risk prediction performance of our model. To test our model from multiple perspectives, we compare the performance of our model with some other networks, traditional CNN, Logistic Regression, SVM and Random Forest. The prediction accuracy and Area Under Receiver Operating Characteristic (AUROC) are used to evaluate our model, since the AUROC represents the classification capability of the classifier. If our model could get high AUROC, it means that the classification threshold could be flexibly adjusted to meet with different needs.

We also test the performance of early prediction. By training the model using the observation data ahead of certain days (90/180 days before), and then test the model using the according dataset. In this way, we can measure the ability of our model to confirm the disease in advance. If our model could get good performance in early prediction, it can be used to assist doctors make timely decision and then give advance treatment for patients.

*4) **Model Tuning**:* To discover the greatest potential of our model, we use a lot of methods to tune our network, such as changing the dropout rate, the early stop condition, the L2 regression and learning rate. For different diseases prediction, the final parameters may not be the same, so the final models for each disease are trained and tested separately. The tuning process and the related control trials will be shown in the latter part of our paper.

C. Experimental Results

In this part, we take classification accuracy (Accuracy) and the area under receiver operating characteristic curve (AUROC) to compare the performance of all methods in two risk prediction tasks. Furthermore, we investigate different hold-off period of two diseases to explore early prediction performance in the proposed framework.

*1) **Risk Prediction Comparison**:* We apply the 3D CNN-SPP and the learned embeddings to predict two important diagnoses, congestive heart failure and diabetes, as two binary classification tasks respectively. We compare 3D CNN-SPP with other baseline models described below:

- **convolutional neural network (CNN)**: The CNN is composed of three layers. The first layer is a one-dimensional convolution layer that captures features over the temporal dimension from input. The second layer is a max-pooling layer along the temporal dimension to keep the most important features across the time. The third layer is a fully connection softmax prediction layer [14].
- **logistic regression (LR)**: The model is used for binary classification to predict whether the diabetes or heart failure is risky for the corresponding patient. We apply L2 regularizers in LR to improve its performance.
- **linear support vector machine (SVM)**: The SVM model represents the examples as points mapped in a space, and then divide the examples with separate categories by a

TABLE I
PREDICTION PERFORMANCE COMPARISON OF 3D CNN-SPP AND BASELINES

Method	Input	Heart Failure		Diabetes	
		Accuracy	AUROC	Accuracy	AUROC
3D CNN-SPP	W2v	**0.8901**	**0.9431**	**0.9955**	**0.9999**
CNN	W2v	0.8630	0.9329	0.9844	0.9989
LR	W2v	0.8625	0.9289	0.9266	0.9802
SVM	W2v	0.8476	0.9140	0.9148	0.9753
RF	W2v	0.8650	0.9269	0.8955	0.9654

TABLE II
EARLY PREDICTION RESULTS OF 3D CNN-SPP

	Heart Failure			Diabetes		
	# of Case	Accuracy	AUROC	# of Case	Accuracy	AUROC
0 days	2660	0.8901	0.9431	2380	0.9955	0.9999
90 days	1806	0.8823	0.9037	1763	0.9949	0.9998
180 days	1780	0.8750	0.9122	1648	0.9395	0.9862

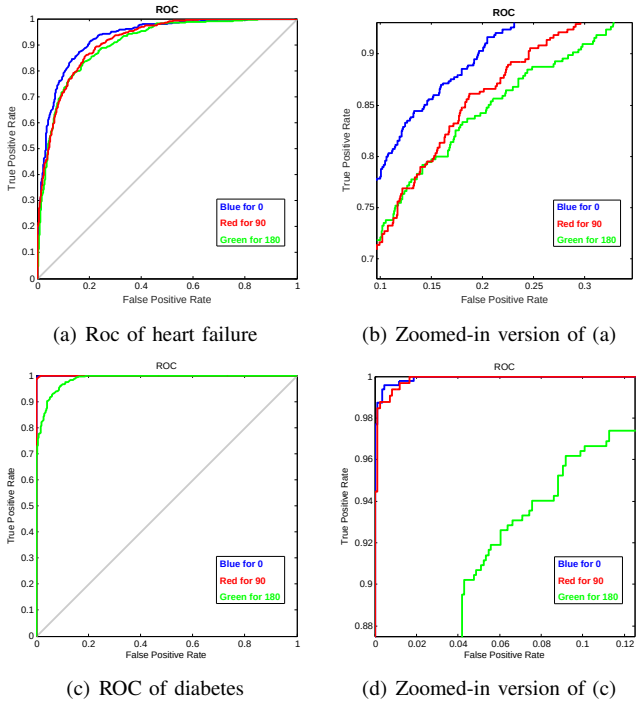
clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall. L2 regularizers are applied in SVM as well.

- **random forest (RF)**: Random forest runs fast, which operates by building a multitude of decision trees at training time and outputting the risk prediction results. We use early stopping to RF for at most 50 trees.

We only use the learned embeddings as input data to evaluate the baselines since it has been demonstrated that learned embeddings get better performance than raw data inputs [14]. For the word2vec embedding input (W2v), we take the embeddings of all events in those records. The classification accuracy and AUROC on the two prediction tasks are summarized in TABLE I. It can be observed that the prediction performance on diabetes is superior to that on heart failure consistently for all models, as the medical data with diabetes may contains more desired details compared with heart failure, which suggests that collecting and keeping more desired medical details are of great significance. For heart failure, LR and RF achieve competitive performance, while SVM achieves slightly worse performance than the former two methods. As for diabetes, SVM outperforms RF, however LR is even better than SVM. For both scenarios, CNN obtains great performance improvements compared to LR, SVM and RF. Nevertheless, the proposed 3D CNN-SPP model achieves prominent predictive accuracy, outperforming all models significantly, demonstrating that 3D CNN model is very effective in boosting the performance on risk prediction tasks. The reason is that 3D CNN extends one more dimension for temporal information by aggregating events into phrases. Because of this construction, 3D CNN can capture more features among the adjacent events and therefore goes a further step compared with 2D CNN. On this stage, it's naturally to consider what impact will be made by different length of phrase. We will have a discussion about it in section IV-D3.

*2) **Early Prediction**:* It is of great significance to make superior prediction in advance. That achievement means the diseases can be diagnosed in their early stage so that timely decisions can be made by doctors. For each patient in the case group, we just take the observation data with a hold-off period (90/180 days) ahead of the time when the first target diagnosis is confirmed to the patient. We extract the patients who have enough records before the first diagnosis from the observation to predict whether the patient will get the target diagnosis in 90/180 days or not. TABLE II shows the amount of the cases we take from the dataset and the performance of each situation. It is a matter of course that the period of time we select earlier, the problem we facing will be more difficult which makes the accuracy decrease. But it's still a prominent performance compared to baselines in TABLE I. For instance, the 180-days early prediction on heart failure achieves accuracy of 0.8750, which is even higher than the best baseline of full observation, whose accuracy reaches 0.8630 by CNN model. The same condition occurs in the 90-days early prediction on diabetes. Note that cases of the patients are decreasing along with the increasing hold-off period, which makes the problem more difficult and leads to performance degradation. In particular, when the hold-off period increases from 90 days to 180 days for diabetes, it presents a sharp performance degradation.

We plot the ROC curves and details of two tasks respectively in Fig. 4. We can observe that 0 days task achieves dramatical performance. Especially for heart failure, the performance shows a decreasing order visibly, in turn, are blue curve for 0 days, red curve for 90 days and green curve for 180 days. But for diabetes, 0 days and 90 days tasks both outperform 180 days significantly, furthermore, 0 days task just achieves sightly better performance than 90 days, which can be noticed from 4(d). In addition, it can be observed from TABLE II and Fig. 4 that similar to what shows in TABLE I, diabetes prediction performance is generally better than that of heart failure, which further indicates the importance of more details in medical data.



(a) Roc of heart failure (b) Zoomed-in version of (a)

(c) ROC of diabetes (d) Zoomed-in version of (c)

Fig. 4. ROC curve comparison of three different hold-off periods for two diseases, respectively. (a) depicts 0/90/180 days ROC of heart failure. We zoom (a) to (b), for better visualization. Similarly, (c) shows different ROC of diabetes and (d) shows the details.

D. More Variants of Our Model

For the sake of better performance, we attune parameters for a number of times to explore a satisfactory setting. The cost function, optimizer, depth of network can all influence the performance a lot. Learning algorithms related to 3D convolutional neural networks also involve many bells and whistles, called hyper-parameters. All above changes from model to model. For our experiments, we set the model hyper-parameters in the following way: (1) dropout rate p is 0.5; (2) early stop threshold value is 10; (3) learning rate is initialed to 0.01, and this value is further reduced whenever the validation error stops improving; (4) loss function is log loss function and (5) the parameter of L2 regression λ is set as 0.001. Besides, all the pooling layers apply max pooling method, and other trainable parameters in this model are initialized randomly.

Here we focus on mini-batch size, network depth and the phrase length, observing the outcomes after we alternate these parameters. More details are shown as following.

*1) **Various mini-batch sizes**:* The mini-batch size n is typically chosen between 1 and a few hundreds, i.e. we set $n = 10$ as first attempt, the outcome achieves an amazing accuracy of 0.9857 for diabetes and 0.8810 for heart failure, even outperforming the previous record in TABLE I of CNN, which reaches accuracy of 0.9844 and 0.8630 for diabetes and heart failure separately. Moreover, it's not the best performance of the network yet. The trend of model behaviors as the mini-batch size increases are depicted in Fig. 5. It is notable that the two tasks have the similar trend when mini-batch size changes.

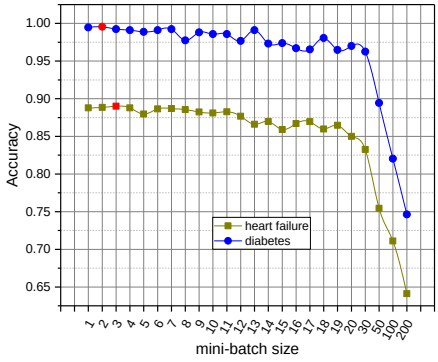


Fig. 5. Here is a trend graph as the mini-batch size increases. Obviously, small mini-batch size shows a better outcome but takes over tens times longer than the same tasks in big mini-batch size. Even though the big mini-batch size lead to faster computation, it meets a sharp drop when exceeding a certain limit.

As n decreases, both two tasks' outcomes rise steadily and with little volatility, reaching the top at $n = 2$ for diabetes and $n = 3$ for heart failure, which are highlighted with red points in the figure. On the contrary, the increase of n lead to a sharp drop. In addition, the mini-batch rarely influences other hyper-parameters so that the other hyper-parameters can be optimized separately.

The impact of n is mainly computational, larger n yields faster computation (with appropriate implementations) but requires visiting more examples in order to reach the same error, since there are less updates per epoch. In theory, this parameter could impact training time and not so much test performance, but it only works to a certain degree. It's true that slightly increasing n just brings a little poor performance, but when it comes to $n = 50$ over and above, the outcome drops lower than 0.9 (for diabetes) or 0.8 (for heart failure). Conversely, small mini-batch size achieves remarkable results. However, it takes so much time to train and converge for small mini-batch sizes. For instance, in diabetes task, $n = 2$ takes 10 even 15 times longer than the time in $n = 10$. Even if it takes 10 times as long to do the mini-batch update, it still seems likely to be better with small mini-batch size, because we update the model more frequently. With these factors in mind, choosing the best mini-batch size is a compromise. Too small, we can't make full use of the benefits of good matrix libraries optimized for fast hardware. Too large, we are simply not updating our weights often enough. What we need in practical application is to choose a compromise value which maximizes the speed of learning with enough precision.

*2) **The Depth of Network**:* In the first set of experiments, we report the performance of the 3D CNN-SPP architecture with various mini-batch sizes described in Fig. 5. As the mini-batch is relatively independent to other hyper-parameters, empirically, once n is selected, it can be generally fixed while the other hyper-parameters can be further optimized. Thereby, we choose n as 2 for diabetes task and 3 for heart failure task

TABLE III
CONTRAST OF DIFFERENT DEPTH OF THE NETWORK.WE CHOOSE THREE SETTING OF CONVOLUTION LAYERS HERE AND SHOW THE ACCURACY FOR DIRECT COMPARASION

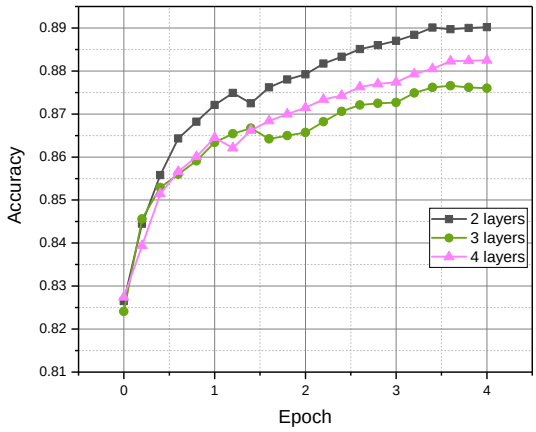
Algorithm	Depth	Heart Failure Accuracy	Diabetes Accuracy
3D CNN-SPP	2 layers	**0.8901**	**0.9955**
3D CNN-SPP	3 layers	0.8751	0.9948
3D CNN-SPP	4 layers	0.8821	0.9903

in the following discussion.

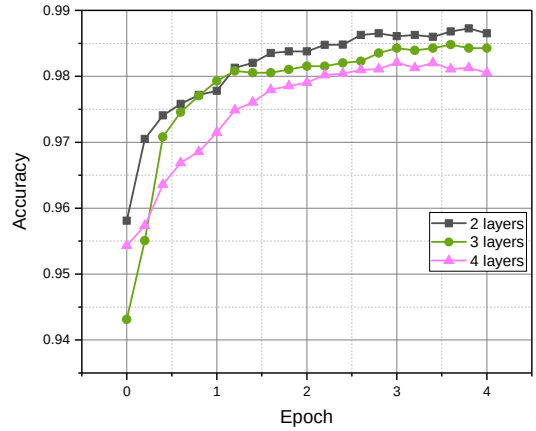
As described in section III-C3, our 3D CNN-SPP model is composed of two 3D phrase-level convolution layers, two 3D phrase-level pooling layers, and a full connection softmax prediction layer in the end of the network. Since the network depth means significantly to performance, here we discuss what influence it will brings when the network depth varies and why we select 2 convolution layers as our choice.

We denote the the number of convolution layers as d here. The various settings of convolutional layers and convergence behaviors during training are plotted in Fig. 6. TABLE III shows the corresponding outcomes for direct comparison. It can be observed that both two tasks perform best when the model has 2 convolution layers. Theoretically, there are no guarantees that increasing the amount of convolution layers will yield better results. In accordance with what we perceive in the experiments, the deeper-layers model presents lower performance on account of having more difficulties to attune the parameters in the pooling layers. The deeper the network is, the harder for us to adjust the pooling rates. Since the spatial pyramid pooling divides the sentences into several length sections for respective pooling processing, parameters in each layer have more difficulties to be fine tuned with the layer getting deeper. Thus, models of 3 or 4 convolution layers could hardly reach a satisfying adjustment.

In addition, the 3 and 4 convolution layers model show lower convergence speed in the later period of the optimization. That might because the deeper network results in more costs on computation. Besides the fact as mentioned above, the 3 or 4 convolution layers model couldn't catch up with the accuracy that 2 layers model shows, but backfire for effect of overfitting. It yields great predictions on training data but poor on testing data. Since more layers lead to more complicated structures with more parameters, the model degrades its generalization performance. Increasing the amount of training data is one way of reducing overfitting. As the number of examples increases, so long as capacity is limited (the number of parameters is small compared to the number of examples), training error and generalization error approach each other. Additionally, we apply L2 regularizers, early stopping and dropout in our architecture to smooth weights and avoid overfitting. It truly improves the performance of 3 and 4 convolution layers model for several percentage points, but there is still a little gap compared to 2 layers model in accuracy and efficiency.



(a) The performance of heart failure



(b) The performance of diabetes

Fig. 6. Here is how the depth of our model influences the eventual accuracy. We display the convergence behaviors during training with d chose from $\{2,3,4\}$. (a) and (b) shows the performance of heart failure and diabetes respectively, both of them demonstrating that model with 2 convolution layers has superior performance.

*3) **Different Phrase Length**:* We still represent the phrase length with k as section III-C1 above. That is to say, k stands for the number of medical events within one phrase. We fix the amount of convolution layers which is represented as d above to 2 so as to exclude interference factors, since $d = 2$ has shown promising performance over the other two models and hardly interacts with k. As shown in TABLE IV, we compare the k with different length $\{1, 2, 3, 4\}$. The $k = 1$ in fact is the original medical events sequence without aggregating into phrases. The outcomes present that the model with aggregated-phrases input data outperforms the raw sequence since it is able to capture more correlations between the events. Particularly, it performs best when $k = 3$. The performance comparison of the model with different length for two tasks is plotted in Fig. 7, for better visualization. We can observe that the two tasks have the similar trend when k varies, which reach the peak at $k = 3$. It's worth mentioning that the diabetes prediction outperforms that in heart failure's task consistently, which once again indicates there are more

details in the medical data of diabetes.

TABLE IV
COMPARISON OF DIFFERENT PHRASE LENGTH $k \in \{1, 2, 3, 4\}$. NOTE THAT WE DISPLAY THE ACCURACY FOR DIRECT COMPARISON

Algorithm	Phrase length	Heart Failure Accuracy	Diabetes Accuracy
3D CNN-SPP	$k = 1$	0.8689	0.9820
3D CNN-SPP	$k = 2$	0.8784	0.9895
3D CNN-SPP	$k = 3$	**0.8901**	**0.9955**
3D CNN-SPP	$k = 4$	0.8780	0.9888

We analyze the results in three aspects. Firstly, the various phrase lengths cause the various input lengths of phrase number, which is represented as l in section III-C2. For instance, setting $k = 3$ will lead the l to $\{98, 142, 186, 246\}$, while we set $k = 2$, it comes to $\{99, 143, 187, 247\}$, and so forth. The factor above further impacts the parameters tuning in pooling layers. Since the general rule should be that the pooling rates in the first few pooling layers should not be too big, like no more than 3 for first pooling layer and no more than 5 for the second and we have to get the fixed-length output through a certain amount of layers, to a certain degree, only when $k = 3$ can we get a satisfying adjustment throughout the entire pooling layers and it greatly improves the performance of model. As we aim to extract the most important features of the input, the loss of the defective pooling could decrease the final results. Secondly, our initial purpose to introduce k is to use a concatenation of k-event-phrases rather than a sequential of medical events as input, so that we could capture features not only inside each phrase, but also between the phrases. It turns out that $k = 3$ shows the best outcome, that is, 3-length-phrase could get the most significant correlations for classification. For further discussion, 2-length phrase could hardly cover enough relevances between adjacent events. For the 4-length phrase, it might exceed the threshold which makes the relevance in a reasonable range, so it becomes just the opposite to our wishes. In other words, the point is, we should make the best use of the features extracted from input data, but avoid making them redundant. Thirdly, since the observation of diagnoses is a time series, the distance between the events must possess the characteristics of temporal correlations. And the longer distance between the two events, the less correlation they could share. In this sense, the phrase length is equivalent to the distance we desire. It's getting closer to optimal solution when $k = 3$ and it's safe to speculate that neither too long nor too short is appropriate.

As a summary, in addition to the hyper-parameters initialed in the beginning, the mini-batch size, depth of the architecture and the phrase length with appropriate values ($n = 2$, $d = 2$ and $k = 3$ as shown above) can boost the performance. Since we mainly evaluate a limited number of hyper-parameters as listed above, we still do not know what influence it will have when we change other hyper-parameters, which is worth further exploring.

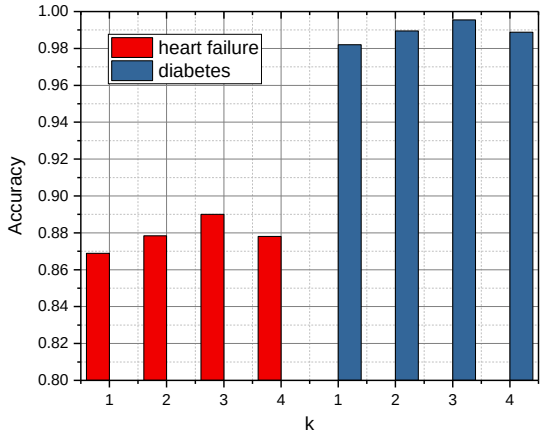


Fig. 7. We display the performance of different phrase length of the two tasks, respectively, for better visualization.

V. DISCUSSION

Our network, together with related works done by other groups, shows that deep learning network can be widely used in medical risk prediction field. Compared with traditional ways of risk prediction, our network has several obvious advantages, the high AUROC and excellent prediction accuracy. In addition, our network can be extended to be applied on the patients with longer or shorter medical records (in our work, the event numbers are limited between 50 and 250) by adjusting some parameters in each layer. What's more, the good performance in early predication 90 or 180 days before shows that our network can be tuned to match specific medical requirements, such as deciding whether to give advance medical treatment or not.

There are still some limitations in our network. Firstly, the EHRs, though containing massive useful information, still need to be pretreated by several steps before sent to our 3D CNN-SPP system. The work of pretreatment is not very convenient and should be handled by professional, which may become a obstacle for putting our model into practical use. In addition, since the training process needs thousands of patient's data to guarantee the prediction performance, our method is difficult to be applied on some rare diseases, where enough EHR records are hard to be gained. In the end, because the details about learning process are hidden, many features used by the algorithm are unknown to us. If we can find a way to witness and understand the features captured by deep learning network, the related medical researches could be greatly promoted.

From another perspective, our method shows that the usage of 3D CNN should not be limited in behavior recognition. What makes 3D CNN powerful is that it can extract information from multi-dimensions, resulting in fully making use of hidden characteristics to make prediction. In addition to disease prediction, it has been proved that 3D CNN could be successfully applied on sentence classification and some similar works. In the future, we believe 3D CNN will be

widely used in different scenarios to update or replace existing models, guaranteeing better classification ability and accuracy.

VI. Conclusion

We developed a 3D CNN-SPP model for risk prediction in this paper. The model extracts features from both spatial and temporal dimensions by performing 3D convolutions and 3D poolings alternatingly. We also introduced SPP to process multi-sized input records to solve the problem of length variety largely. We found that embedding the EHRs data into vectors and splitting the original records of each patient into phrases are very effective in boosting the performance. We evaluated the 3D CNN-SPP models on the pretreated data and demonstrate its superior performance to the state-of-the-art models and make great early predictions. In addition, a series of contrast tests showed that the model with 2 convolution layers and small mini-batch can lead to the best result.

It would be interesting to exploit the underlying information of the features throughout the whole algorithm and it's worth to explore the unsupervised training method in the future. Applying 3D CNN-SPP model in other fields and scenarios are also promising.

References

[1] L. B. Madsen, "Data-driven healthcare: How analytics and bi are transforming the industry," *Wiley*, 2014.

[2] Z. P. Liu, L. Y. Wu, Y. Wang, X. S. Zhang, and L. Chen, "Prediction of protein-rna binding sites by a random forest method with combined features," *Bioinformatics*, vol. 26, no. 13, pp. 1616–1622, 2010.

[3] W. Raghupathi and V. Raghupathi, "Big data analyticsarchitectures, implementation methodology, and tools," *Big Data, Mining, and Analytics*, 2014.

[4] S. Kleinberg and G. Hripcsak, "A review of causal inference for biomedical informatics," *Journal of Biomedical Informatics*, vol. 44, no. 6, pp. 1102–1112, 2011.

[5] J. Zhou, F. Wang, J. Hu, and J. Ye, "From micro to macro: Data driven phenotyping by densification of longitudinal electronic medical records," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '14. New York, NY, USA: ACM, 2014, pp. 135–144. [Online]. Available: http://doi.acm.org/10.1145/2623330.2623711

[6] J. C. Ho, J. Ghosh, and J. Sun, "Marble: High-throughput phenotyping from electronic health records via sparse nonnegative tensor factorization," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '14. New York, NY, USA: ACM, 2014, pp. 115–124. [Online]. Available: http://doi.acm.org/10.1145/2623330.2623658

[7] F. Wang, N. Lee, J. Hu, J. Sun, and S. Ebadollahi, "Towards heterogeneous temporal clinical event pattern discovery: A convolutional approach," in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '12. New York, NY, USA: ACM, 2012, pp. 453–461. [Online]. Available: http://doi.acm.org/10.1145/2339530.2339605

[8] Z. Che, D. Kale, W. Li, M. T. Bahadori, and Y. Liu, "Deep computational phenotyping," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '15. New York, NY, USA: ACM, 2015, pp. 507–516. [Online]. Available: http://doi.acm.org/10.1145/2783258.2783365

[9] R. Miotto, L. Li, B. A. Kidd, and J. T. Dudley, "Deep patient: An unsupervised representation to predict the future of patients from the electronic health records," *Scientific Reports*, vol. 6, pp. 26 094+, May 2016. [Online]. Available: http://dx.doi.org/10.1038/srep26094

[10] Z. C. Lipton, D. C. Kale, C. Elkan, and R. C. Wetzel, "Learning to diagnose with lstm recurrent neural networks." *CoRR*, 2015.

[11] E. Choi, M. T. Bahadori, E. Searles, C. Coffey, M. Thompson, J. Bost, J. Tejedor-Sojo, and J. Sun, "Multi-layer representation learning for medical concepts," in *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16, 2016, pp. 1495–1504. [Online]. Available: http://doi.acm.org/10.1145/2939672.2939823

[12] S. Ghosh, Y. Cheng, and Z. Sun, "Deep state space models for computational phenotyping," in *2016 IEEE International Conference on Healthcare Informatics, ICHI 2016, Chicago, IL, USA, October 4-7, 2016*, 2016, pp. 399–402. [Online]. Available: http://dx.doi.org/10.1109/ICHI.2016.71

[13] Y. Cheng, F. Wang, P. Zhang, and J. Hu, "Risk prediction with electronic health records: A deep learning approach," in *Proceedings of the 2016 SIAM International Conference on Data Mining, Miami, Florida, USA, May 5-7, 2016*, 2016, pp. 432–440. [Online]. Available: http://dx.doi.org/10.1137/1.9781611974348.49

[14] Z. Che, Y. Cheng, Z. Sun, and Y. Liu, "Exploiting convolutional neural network for risk prediction with medical feature embedding," *CoRR*, vol. abs/1701.07474, 2017. [Online]. Available: http://arxiv.org/abs/1701.07474

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition technical report," *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, vol. 37, pp. 1904 – 1916, 2015.

[16] G. Hripcsak and D. J. Albers, "Next-generation phenotyping of electronic health records," *Journal of the American Medical Informatics Association Jamia*, vol. 20, no. 1, p. 117, 2013.

[17] R. Amarasingham, B. J. Moore, Y. P. Tabak, M. H. Drazner, C. A. Clark *et al.*, "An automated model to identify heart failure patients at risk for 30-day readmission or death using electronic medical record data," *Medical Care*, vol. 48, no. 11, pp. 981–988, 2010.

[18] J. C. Ho, J. Ghosh, S. R. Steinhubl, W. F. Stewart, J. C. Denny *et al.*, "Limestone: High-throughput candidate phenotype generation via tensor factorization," *Journal of Biomedical Informatics*, vol. 52, pp. 199–211, 2014.

[19] D. Gotz, F. Wang, and A. Perer, "A methodology for interactive mining and visual analysis of clinical event patterns using electronic health record data," *Journal of Biomedical Informatics*, vol. 48, no. C, pp. 148–159, 2014.

[20] F. Wang, N. Lee, J. Hu, J. Sun, and S. Ebadollahi, "Towards heterogeneous temporal clinical event pattern discovery: a convolutional approach," *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM*, pp. 453–461, 2012.

[21] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[22] M. I. Jordan and M. T. M, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, 2015.

[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *International Conference on Neural Information Processing Systems, Curran Associates Inc.*, pp. 1097–1105, 2012.

[24] S. Ji, W. Xu, M. Yang, and k. Yu, "3d convolutional neural networks for human action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, pp. 221–231, 2013.

[25] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research 15*, pp. 1929–1958, 2014.