Event2vec: Learning Representations of Events on Temporal Sequences

Shenda Hong[1,2], Meng Wu[1,2], Hongyan Li[1,2,⋆], and Zhengwu Wu[3]

[1] Key Laboratory of Machine Perception, Ministry of Education, Beijing, China
lihy@cis.pku.edu.cn
[2] School of EECS, Peking University, Beijing, China
[3] Science and Technology on Information Systems Engineering Laboratory, Beijing Institute of Control and Electronic Technology, Beijing, China

Abstract. Sequential data containing series of events with timestamps is commonly used to record status of things in all aspects of life, and is referred to as temporal event sequences. Learning vector representations is a fundamental task of temporal event sequence mining as it is inevitable for further analysis. Temporal event sequences differ from symbol sequences and numerical time series in that each entry is along with a corresponding time stamp and that the entries are usually sparse in time. Therefore, methods either on symbolic sequences such as word2vec, or on numerical time series such as pattern discovery perform unsatisfactorily. In this paper, we propose an algorithm called event2vec that solves these problems. We first present Event Connection Graph to summarize events while taking time into consideration. Then, we conducts a training Sample Generator to get clean and endless data. Finally, we feed these data to embedding neural network to get learned vectors. Experiments on real temporal event sequence data in medical area demonstrate the effectiveness and efficiency of the proposed method. The procedure is totally unsupervised without the help of expert knowledge. Thus can be used to improve the quality of health-care without any additional burden.

Keywords: Temporal Event Sequences, Learning Representations

1 Introduction

Sequential data containing series of events with timestamps is referred to as temporal event sequences [9]. Temporal event sequences is commonly used to record status of things in all aspects of life, such as customer purchase sequences, motion gesture sequences, and hospital treatment sequences. On the left side of Fig.1 shows the daily treatment procedures of patients in hospital [13]. Each ITEM is an event, and (ITEM, TIMESTAMP) is a temporal event. Intuitively, this temporal event sequence tells a detailed story about how patients behave in the hospital. And that story is written in the language of events.

⋆ Corresponding author.

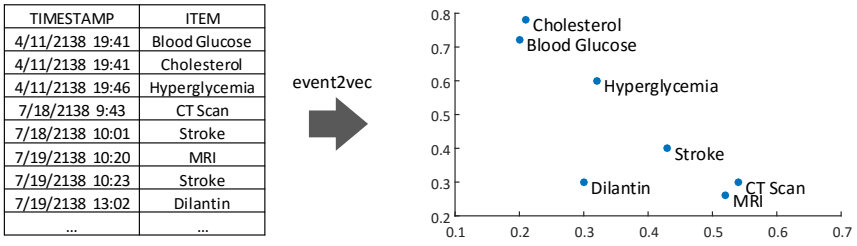

Fig. 1. (Left) An example of a temporal event sequence from EHR. The first column is TIMESTAMP, indicating the occurring time, while the second column is ITEM, indicating the event recorded during the treatment. (Right) Event2vec transforms symbolic events into numerical vectors (take two-dimension vectors for example).

Consider the task of data mining on temporal event sequence, it would be a fundamental step to transform symbolic events into numerical vectors (Fig.1 right). Thus, learning representations has good prospect of applications such as healthcare [2][18], marketing analytics [9] and motion recognition [5]. On the one hand, the representation itself provides an insight into connection between different events. On the other hand, the representation can be used for practical tasks such as recommendation and clustering. In the above example, the records of one patient require systematic analysis from doctors to provide individualized diagnosis and treatment. It is critical in scenarios like clinical reasoning (what comes the most probability when Cholesterol appears), complication forecasting (what symptoms are Hyperglycemia complication). However, the task would be extremely difficult to fulfill manually even for experts, since the events are numerous and are usually from various medical fields.

Although some recent researches focusing on data mining from temporal event sequences have made certain achievements on learning representations of events. It is still an open challenge due to the following problems.

- **Complexity** : Symbolic methods usually use a combination of symbols as representations (referred to as patterns) [1][15]. Whereas the number of all possible patterns would be 2^n, where n is the number of distinct events. Although recent works have improved the computing efficiency dramatically, the number of patterns still explodes along with the increasing of item quantity [3], which leads to the complexity of mined representations.
- **Sparsity** : Numerical methods are based on continuous numeric sequences [7][10] [21][18]. If we want to apply these methods on temporal event sequences, they should first be transformed into continuous numeric sequences (one-hot encoding for example). However, the transformation procedure will lead to high sparsity and introduce lots of noise.
- **Unbalance** : Some events rarely appear but are very important (such as “vital sign ”in EHRs). Meanwhile, not all of the potential patterns have been appeared in orig-

inal data. Both symbolic methods and numerical methods are unsensitive to these important but rare events. [1][15][10][21][18].
- **Time Consideration** : Recent learning representation works implemented neural network architecture like word2vec [11][12], and they did get more dense vectors and resolved some of the above problems. However, they did not take timestamp into consideration since there is no time information in text data. It is a big loss to ignore the occurring time which contains rich semantic information. For example, consider two events, one next to each other with large time interval between. We shouldn't treat them as adjacent events at all as they are irrelevant at high rates, but word2vec has no mechanism to handle that problem.

In this paper, we propose an algorithm called event2vec to embed temporal events into the vector space based on sequential data. Event2vec takes time information into consideration, while implementing neural network architecture, which makes the method benefit from both timestamp information and distributed representation. In particular, event2vec has the following properties:

- Event2vec summarizes large scale temporal event sequence to an event connection graph, with nodes representing events, edges representing relations extracted from temporal sequence. It considers time relationship of events.
- Event2vec conducts a training sample generator. We can relieve unbalance and sparsity problems with the help of the sample generator.
- Event2vec trains an embedding neural network, and represents events in a distributed way. It avoids exaggerated complexity by adjusting the number of units in hidden layer, while keeping the internal relationship among events.
- Event2vec is a totally unsupervised algorithm with no need of expert knowledge, and it can discover relations between events automatically.

We conduct comprehensive experiments on real temporal event sequence data. We unsupervisedly discover implicit relations between various kinds of events. Experiments demonstrate the effectiveness and efficiency of our proposed method. Specifically, we implement event2vec on electronic health records (EHRs) data, and map clinical events to vector space, which reveals some interesting results. The procedure is totally unsupervised. Thus can be used to improve the quality of health-care.

2 Preliminaries

In this section, we introduce our notations and some definitions. Notations are shown in Table 2.

Definition 1. *(Event Set): Event Set Eve is a collection that records all possible conditions of something that would happens. And we use N to denote the number of distinct events in Event Set Eve. For example, the event set of signal light in cross road should be $Eve = \{red, green, yellow\}$, and $N = 3$.*

Definition 2. *(Temporal Event): A temporal event is represented as a two-gram tuple (e, t), where e is the event, t is the occurring time. $e \in Eve$.*

Definition 3. *(Temporal Event Sequence): Given a set of N events Eve, $S = \{(e_i, t_i)|e_i \in Eve, \}$ denotes a temporal event sequence.*

Symbol	Description
S	Temporal event sequences dataset
Eve	Set of all possible events
N	Number of distinct events
G	Event connection graph
T	Time threshold (Hyper-parameter in Constructing Graph Process)
Δ	Shrink coefficient (Hyper-parameter in Constructing Graph Process)
$Corpus$	Generated data for embedding
Φ	Matrix of event representations
d	Embedding dimension (Hyper-parameter in Embedding Process)
N_w	Number of events generated in a sequence (Hyper-parameter in Embedding Process)
N_l	Number of samples in generated data (Hyper-parameter in Embedding Process)

Table 1. Notations and Meanings

In the following section, we will introduce the details of how event2vec learns representations of symbolic events while considering timestamp information.

3 Method

In this section, we present the detailed procedure of the event2vec algorithm. The framework of our method is shown in Algorithm 1. First of all, for the purpose of taking timestamps into consideration, we summarize temporal event sequences into a single graph called event connection graph, with nodes representing events and edges representing extracted relations. A training sample generator is then conducted with reference to the idea of probabilistic walk. It generates a mini-batch of clean and endless $Corpus$ once a time. Finally, we feed these data to embedding neural network and output learned vectors.

Algorithm 1 Event2vec

1: **Input**: $S, Eve, d, \Delta, T, N_w, N_l$
2: **Output**: vectors of event representations Φ
3: G = ConstructingGraph(S, Eve, Δ, T)
4: **while** $iter < N_l$ **do**
5: $Corpus$ = SampleGenerator(G, Eve, N_w)
6: LearningRepresentations($Corpus$, Φ, d)
7: $iter$ = $iter$ + 1
8: **end while**

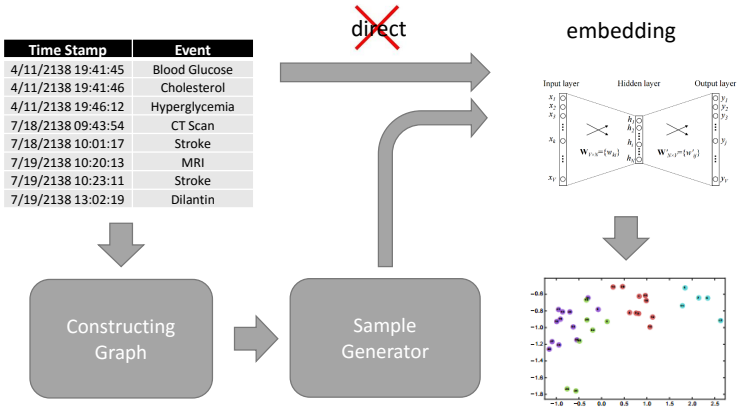

Fig. 2. Framework of event2vec

3.1 Constructing Event Connection Graph

When realizing the idea, a natural solution would be to apply the word2vec directly by treating the sequences of events as the sequences of words. However, it would be difficult to deal with the challenges introduced in section 1 if using word2vec directly. Hence, for the purpose of taking timestamps into consideration, we summarize the temporal event sequence into a compact graph. In this way, the following purposes would be accomplished: Firstly, extracting temporal information of events. Secondly, condensing the sparse data and settling the problems of unbalance at the same time. Lastly, simplifying the task of updating edge weight with incoming data, which can be updated incrementally instead of completing the entire re-computation.

The comparison between word2vec and event2vec in the experiment section shows the effectiveness of constructed graph. Now we introduce the process of constructing event connection graph.

Definition 4. *(Event Connection Graph): Let $G =< V, E >$ be a directed weighted graph constructed from S, where each vertex e_i in V represents an unique event from Eve and edges in E represent relations extracted from temporal event sequence. Weight of the directed edge from node e_i to node e_j is calculated by:*

$$G_{ij} = \sum_{1 \le i < j \le N} 1\{S(t_1) = e_i\} \wedge 1\{S(t_2) = e_j\}\delta(t_2 - t_1) \tag{1}$$

where $1\{\cdot\}$ is set to 1 if its argument is true, and $\delta(x)$ is a function mapping time interval to the relation measurement of two events.

Note that $\delta(x)$ is non-increasing, and the function satisfies $\delta(0) = 1$ and $\delta(T) = 0$. The relation measurement decreases to zero smoothly with the increase of time interval until reaching the threshold T.

$$\delta(x) = \begin{cases} exp(-x/\Delta), & 0 \le x < T \\ 0, & otherwise \end{cases} \tag{2}$$

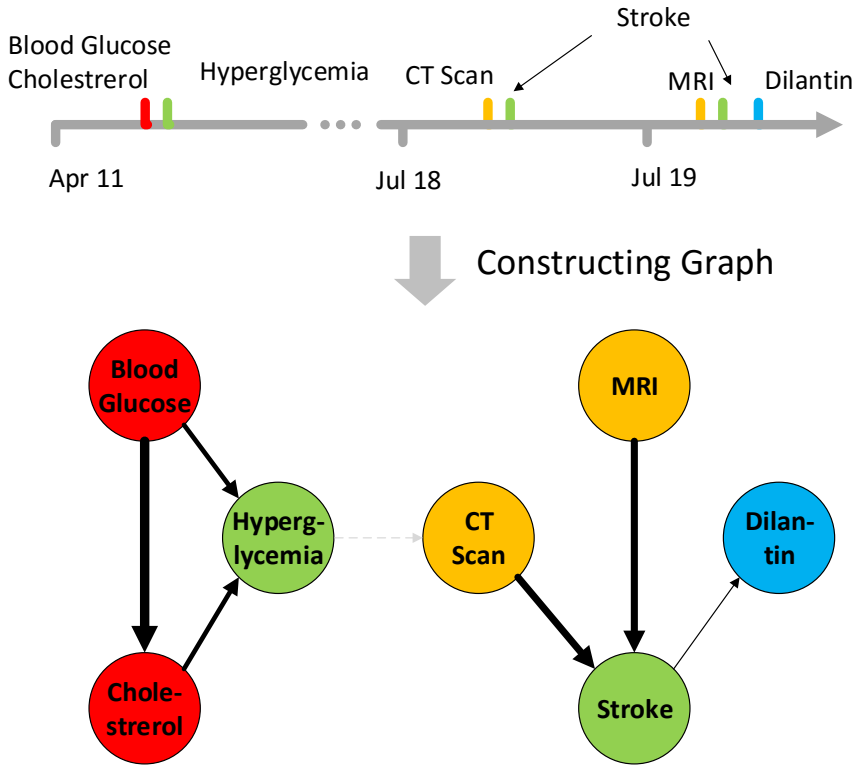

Fig. 3. The process of constructing event connection graph. Each vertex in V represents an unique event from Eve, and edges in E represent relations extracted from temporal event sequence. Weight between the node i to node j is calculated by Equation.1

Algorithm 2 ConstructingGraph

Input: S, Eve, Δ, T
Output: event connection graph G
for i = 1 to Length(S) **do**
  (e_i, t_i) = S(i)
  for j = i to Length(S) **do**
    (e_j, t_j) = S(j)
    pos_i = Eve.Index(e_i)
    pos_j = Eve.Index(e_j)
    $G[pos_i, pos_j] += \delta(t_i, t_j, \Delta, T)$
  end for
end for

3.2 Sample Generator

The next step is to get samples for the training process. It is beneficial to use Probabilistic Walk for training instead of original temporal event sequence. On the one hand, it relieves unbalance and sparsity effectively. even if an event appears only once, it must connect with other events with a small probability. So we can resample to get more training data containing that event. On the other hand, it can generate endless training data to tackle the problem of lack of original training data and guarantee the convergence of training process. The experiment in Section 4.3, shows the effectiveness of sample generator.

We incorporate Probabilistic Walk into the previously constructed event graph. It is a stochastic process starting with a single randomly chosen event e_i. And one of the neighbors of e_i denoted as e_j would be added based on a probability in direct proportion to the edge weight between e_i and e_j. The probability of choosing event e_j after event e_i is

$$P(i \to j) = W_{ij} / \sum_i W_{ij} \tag{3}$$

The sample generator generates a mini-batch of $Corpus$ at a time.

3.3 Learning Presentations

By now, we can get any number of event sequences via Sample Generator. Although these event sequences have no timestamps, they still imply time information thanks to the event Connection Graph. The remaining challenge is to learn representations from the generated sequences.

Formally, given a generated sequence of $Corpus = \{e_1, e_2, e_3, ..., e_n\}$, we would like to predict events nearby a certain event e_i ($e_i \in V$) That is, to maximize the conditional probability of a center event within a context with width L in the training cases. The probability of nearby events $e_{c-L}, ..., e_{c+L}$ given the center event e_i is written as:

$$Pr(\{e_{c-L}, ..., e_{c+L}\}|e_c) = \prod_{k=c-L}^{c+L} Pr(e_k|\Phi(e_c)) \tag{4}$$

Algorithm 3 SampleGenerator

Input: G, Eve, N_w
Output: a mini-batch of training data $Corpus$
Empty($Corpus$)
e_c = Random(Eve)
$Corpus$.Append(e_c)
$Distri_{e_c} = G[e_c]$
for i = 1 to $numBatch$ **do**
  for j = 1 to N_w **do**
    e_{next} = Walk($Eve, Distri_{e_c}$)
    $Corpus$.Append(e_{next})
  end for
end for

Where Φ is our desired representations. After transforming the expression into log-likelihood, the overall loss function for the optimization process would be:

$$J(\Phi) = -\sum_{c=1}^{n} \sum_{k=c-L}^{c+L} logPr(e_k|\Phi(e_c)) \tag{5}$$

We noticed that this loss function is exactly the same as that of SkipGram model in word2vec. And we can use the standard optimization method to solve Eq.5, namely stochastic gradient descent (SGD). The training procedure can be endless on the basis of mini-batches generated from generator. We first randomly initialize representation vector Φ. On each iteration, we map each event e_k to its current representation vector, and then maximize the probability of its neighbors in the sampled sequence.

Algorithm 4 Learning Representations

Input: $Corpus$, Φ, d
Output: vectors of events representations Φ
for $sequence_{e_c} \in Corpus$ **do**
  for $e_k \in sequence_{e_c}$ **do**
    $J(\Phi) = -logPr(e_k|\Phi(e_c))$
    $\Phi = \Phi - \alpha * \frac{\partial J}{\partial \Phi}$
  end for
end for

4 Experiments

4.1 Experimental Setup

In consideration of realistic application prospect, we use MIMIC (Medical Information Mart for Intensive Care) — a real world temporal event sequence dataset, which

is collected on over 58,000 ICU patients at the Beth Israel Deaconess Medical Center (BIDMC) from June 2001 to October 2012 [4]. It contains millions of records of medical events and there are 5373 unique medical events in Eve. Our goal is to learn distributed representation of these events.

We design the following mechanism to evaluate the quality of the learned vectors based on the fact that these medical events inherently belong to five categories: demographics, diagnosis, medications, lab tests and procedures. Thus, aiming to estimate whether the learned vectors of events from different categories are well distinguishable, we use the same cluster method on the learned vectors and compare the clustering results. We use the typical evaluation index of clustering, namely Rand Index [16]. It is defined as:

$$RI = \frac{TP + TN}{TP + TN + FP + FN} \tag{6}$$

where, T stands for true, represents that two events belong to the same cluster, P stands for positive, represents that two events are assigned to the same cluster. Then, TP is the number of event pairs in which two events belong to the same cluster and are assigned to the same cluster as well. TN, FP, FN can be explained as well. Intuitively, RI goes higher if the cluster method correctly assign more pairs of events. Similarly, we also define Precision ($P = \frac{TP}{TP+FP}$) and F_1 score($F_1 = \frac{2TP}{2TP+FP+FN}$) to evaluate the quality of representation.

All tasks are executed on a machine equipped with Intel Core i5, 16GB RAM using Python 2.7.3. The clustering algorithm is K-means, which is realized by scikit-learn 0.18.

4.2 Event2vec Hyper-parameter Analysis

Firstly, we evaluate different groups of Δ, T, N_l, d, and see the influence of these hyper-parameters.

We evaluate Δ, T in graph construction process, and fix other hyper-parameters. As shown in Figure 4, we can see that the method achieves highest RI, Precision and F_1 when Δ is about 300. As Δ increases, RI, Precision and F_1 will increase as first, but decrease after Δ exceeds a certain threshold. There are two main reasons. On the one hand, higher T means that events with longer time interval are considered relevant, which will bring more noise in the connection graph. On the other hand, higher Δ means events with shorter time interval will add more weight to the corresponding edge of the connection graph. As a result, there exists a balance between T and Δ.

Secondly, we evaluate N_l, d with other hyper-parameters fixed. These two parameters are from sample generate and embedding process. N_l controls sample scale input to embedding neural network, or iteration number in other words. d is the number of nodes in hidden layer of embedding neural network. Result is shown in Figure 5. As we can see, RI, Precision and F_1 all increase when N_l goes large, and get steady when $N_l > 50,000$. Besides, RI, Precision and F_1 remains steady when d is larger than 6. It demonstrates that our model achieves good performance even when the vectors are in six-dimension space, which also shows the robustness of our model.

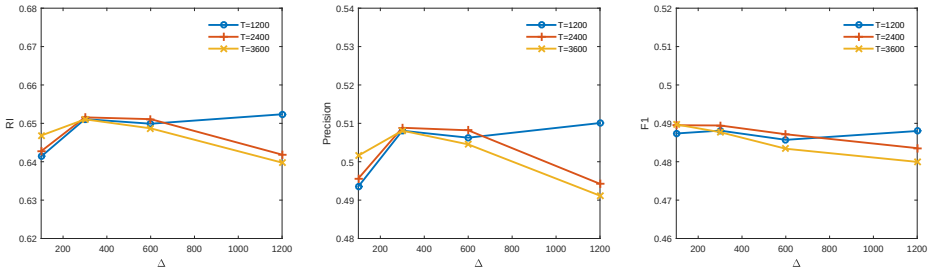

Fig. 4. The effect of hyper-parameters Δ and T. Measured by RI (left), Precision (middle) and F_1 (right).

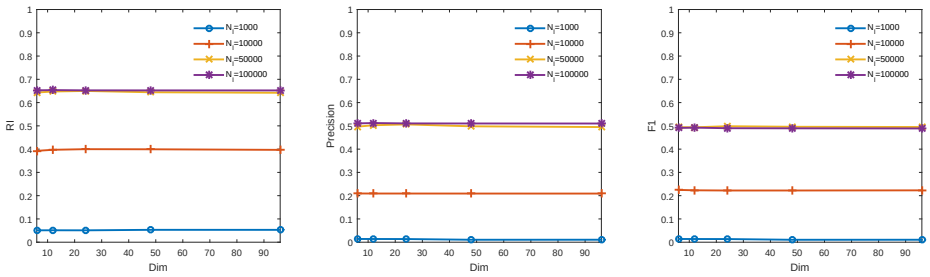

Fig. 5. The effect of hyper-parameters N_l and d measured by RI (left), Precision (middle) and F_1 (right).

4.3 Comparison with Other Methods

As far as we know, there does not exist a general framework for unsupervised learning representation while considering timestamps. We compare our model with the following state-of-the-art methods mentioned before.

- **Count-hot:** For each event, Count-hot constructs a vector using the occurrence number of the event in temporal event sequence of each patient. It means that Count-hot uses each patient to represent one dimension. So the number of dimensions of Count-hot vectors is equal to the number of patients.
- **word2vec:** As introduced in section 3, word2vec applied the skip-gram model directly by treating the sequences of events as the sequences of words. It can be regard as event2vec without the graph construction step.
- **DirectGraph:** After constructing the event graph, DirectGraph regards each column (or row) from the adjacent matrix as a vector. It can be regard as event2vec without the skip-gram embedding step.
- **KNN:** KNN constructs event graph based on their k nearest neighbors. It means that the weight between e_i and e_j is the occurrence number of (e_i, e_j) pair being k nearest neighbors. The remaining procedures of sample generating and embedding process are the same as event2vec. It can be regard as event2vec with a modified graph construction step.

	Count-hot	word2vec	DirectGraph	KNN (k=3)	KNN (k=5)	KNN (k=7)	Event2vec
RI	0.3938	0.5486	0.3932	0.6202	0.6157	0.6210	**0.6516**
Precision	0.3390	0.3863	0.3372	0.4462	0.4458	0.4475	**0.5088**
F_1	0.4667	0.4238	0.4629	0.3586	0.3913	0.3572	**0.4894**

Table 2. Comparison with Different Methods. The result shows that event2vec achieves better effect than other methods on metrics of RI, Precision and F_1.

	Demographics	Diagnosis	Medication	Lab tests	Procedures
Relativity	1.42	1.50	1.33	1.24	1.25

Table 3. Relativity

The event2vec configuration is $\Delta = 300, T = 2400, L = 5$. To make it a fair comparison, we also: (1) Introduce PCA to reduce the dimension of Count-hot and DirectGraph to 48; (2) Set the hidden layer node numbers to 48 in all three methods, namely event2vec, word2vec and KNN; (3) Set N_l to 100,000 to generate the same scale of corpus for word2vec and event2vec. The experiment results are shown in Table 2.

The result shows that on all three metrics, event2vec achieves the highest performance. The only difference between word2vec and event2vec is that, word2vec does not have the procedures of graph construction and sample generation. It shows the effectiveness of these two steps, explanations of which can be found in Section 3.1. Besides, the differences between event2vec and DirectGraph is that, DirectGraph does not have the procedures of random walk and skip-gram, explanations of which can be found in Section 3.2. As for KNN, the whole process also contains graph construction, random walk and skip-gram, but a different schema is used for computing weight of edges in the graph. It demonstrates that our method of graph constructing is better than KNN, after taking timestamps into account.

4.4 Interpretation

Consider the importance of interpretability in healthcare, we conduct the following experiments in collaboration with medical experts, and they confirm the explanation of our learned representations.

We first perform a relativity assessment by randomly selecting 74 events along with their top 10 most similar events. This shows whether the learned representations effectively capture the latent relationships among them. As shown in Table 3, two medical experts checked the result and decided whether an event pair is related, possible or unrelated. Obviously, we can come to the conclusion that all of the relativity scores are greater than 1, which means that event2vec successfully finds the high related events of the given ones.

Furthermore, we list five selected events along with their top 10 most related events in Table 4 A comprehensive example would be that ECG, the measurement of cardiac electrical activity, can be reflected by BP (Blood Pressure).

Seizure	Apnea	ECG	Cholesterol	Intra Cranial Pressure
Ataxia	high min volume	BP Right Leg Diastolic	HDL	SVV Arterial
CIWA Sum Total	high mv	ECG	Fibrinogen	Intra Cranial Pressure
Tactile Disturbances	apnea	BP Right Arm Mean	Triglyceride	Central Venous Pressure
Auditory Disturbance	Inspired Gas Temp	BP Right Leg Systolic	C Reactive Protein CRP	Cerebral Perfusion Pressure
Orient/Clouding Sens	ETT Location	BP Right Leg Mean	Amylase	CO Arterial
Visual Disturbances	ETT Mark	BP Right Arm Systolic	AST	Bladder Pressure
Aggitation	Apnea Time Interval	BP Left Leg Mean	Cholesterol	Transpulmonary Pressure Exp. Hold
Seizure	Ventilator Type	BP Left Arm Diastolic	LDL calculated	Glucose whole blood
Headache	Airway Size	BP Left Arm Mean	Alkaline Phosphate	Arterial Blood Pressure mean
Anxiety	Airway Type	BP Left Arm Systolic	Lipase	Arterial Blood Pressure systolic

Table 4. Five selected events and their top 10 most related events

Dimension 1	Dimension 2	Dimension 3	Dimension 4
Religion	Sodium ApacheIV	WhiteBloodC 4.0-11.0	Ventilator Mode
Highest Level	RRScore ApacheIV	TCO2 (other)	Tracheostomy Cuff
Height (cm)	RR ApacheIV	Red Blood C(3.6-6.2)	Tidal Volume (Set)
Height	OxygenScore ApacheIV	RBC(3.6-6.2)	Return Pressure
Gestational Age	MAP ApacheIV	ph (other)	Plateau Off
Code Status	CreatScore ApacheIV	pH (cap)	Minute Volume Alarm - Low
Birthweight (kg)	BiliScore ApacheIV	pCO2 (other)	Low Exhaled Min Vol
Age	Bilirubin ApacheIV	pCO2 (cap)	Flow By (lpm)
Admission Weight (lbs.)	Apache IV Age	HGB (10.8-15.8)	Cuff Leak
Admission Weight (Kg)	AgeScore ApacheIV	Base Excess (other)	Access Pressure

Table 5. Four selected dimensions along with the top 10 events in the corresponding dimension

In addition, we randomly choose 4 dimensions, and list events with the largest 10 values in the chosen 4 dimensions in Table 5. We can see that the events in Dimension 1 are mostly related to demographics of patients; Dimension 2 are mostly related to Apache score (a commonly used health measurement); Dimension 3 are mostly related to chemical index of blood lab test reports; Dimension 4 are mostly related to clinical procedures by medical devices.

In conclusion, event2vec successfully finds meaningful events. These results demonstrate that event2vec can be used for clinical reasoning, medication recommendation and complication forecasting.

5 Related Work

For a comprehensive review, we introduce learning representation from both continuous sequential data and symbolic sequential data.

Learning representation from continuous sequential data also refers to pattern discovery on sequential data. The methods in this category usually extract a small fraction of sequences such as motif [10], shapelet [21], signature [18], and PLR [17]. [18] proposed a convolutional method that learns temporal event signatures and uses these signatures to learn representations. In addition, [19] using pattern-based hidden Markov model to find dynamics for semantic representation from time series data. We should first transform temporal event sequence to continuous numeric sequence (one hot for example), and then apply numerical methods. However, the data would have high sparsity. Besides, Some events rarely appear but are very important. It is difficult to capture

the micro relationship while keeping the macro information [9] follows [8], propose skeleton of the graph constructed by temporal event sequence, and discover higher granularity representation from original data.

Learning representation from symbolic sequential data usually use a combination of symbols as representations (referred to as patterns) methods usually using a combination of symbols as representations (referred to as) [1][15][6]. Although some efforts to reduce the complexity of pattern sets, they still get the initial patterns and then reduce them by approximation [14] or clustering [20]. and it might lead to a complexity of mined representations. Recently, neural network architecture was proposed for distributed representations for word embedding [11][12], does get more dense vectors and resolve above problems. However, they did not take timestamps into consideration since no time information in text data. It is a big loss to ignore the occur time which provide rich semantic information. [22] use transitive distance to transform categorical data to numerical representations. however, the works mentioned above also suffer from complexity and unbalance.

6 Conclusion and Future Work

In this paper, we propose the event2vec algorithm that transforms large scale symbolic events from sequential data into numerical vectors. Event2vec takes temporal information into consideration, and solves the problem of high complexity, sparsity and unbalance. Concretely, our Event Connection Graph takes time into consideration; Sample Generator solves sparsity and unbalance in raw data, final embedding neural network controls complexity of learned vectors. Experiments show that event2vec outperforms other learning representation methods, while providing good interpretation of events. The procedure is totally unsupervised without the help of expert knowledge. Thus, the algorithm can be used in real-world applications, such as to improve the quality of health-care without any additional burden.

In the future, we plan to modify the embedding neural network, and take the structure information extracted from constructed graph into consideration.

References

1. Agrawal, R., Srikant, R.: Mining sequential patterns. In: Proceedings of the 11th International Conference on Data Engineering (ICDE 1995), March 6-10, 1995, Taipei, Taiwan. pp. 3–14 (1995), http://dx.doi.org/10.1109/ICDE.1995.380415
2. Choi, E., Bahadori, M.T., Searles, E., Coffey, C., Thompson, M., Bost, J., Tejedor-Sojo, J., Sun, J.: Multi-layer representation learning for medical concepts. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016. pp. 1495–1504 (2016), http://doi.acm.org/10.1145/2939672.2939823
3. Fournier-Viger, P., Gomariz, A., Campos, M., Thomas, R.: Fast vertical mining of sequential patterns using co-occurrence information. In: Advances in Knowledge Discovery and Data Mining - 18th Pacific-Asia Conference, PAKDD 2014, Tainan, Taiwan, May 13-16, 2014. Proceedings, Part I. pp. 40–52 (2014), http://dx.doi.org/10.1007/978-3-319-06608-0_4

4. Goldberger, A.L., Amaral, L.A., Glass, L., Hausdorff, J.M., Ivanov, P.C., Mark, R.G., Mietus, J.E., Moody: Physiobank, physiotoolkit, and physionet components of a new research resource for complex physiologic signals. Circulation 101(23), e215–e220 (2000)
5. Jaimes, A., Sebe, N., Boujemaa, N., Gatica-Perez, D., Shamma, D.A., Worring, M., Zimmermann, R. (eds.): ACM Multimedia Conference, MM '13, Barcelona, Spain, October 21-25, 2013. ACM (2013), http://dl.acm.org/citation.cfm?id=2502081
6. Kim, Y., Han, J., Yuan, C.: TOPTRAC: topical trajectory pattern mining. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015. pp. 587–596 (2015), http://doi.acm.org/10.1145/2783258.2783342
7. Lin, J., Keogh, E.J., Lonardi, S., Chiu, B.Y.: A symbolic representation of time series, with implications for streaming algorithms. In: Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery, DMKD 2003, San Diego, California, USA, June 13, 2003. pp. 2–11 (2003), http://doi.acm.org/10.1145/882082.882086
8. Liu, C., Wang, F., Hu, J., Xiong, H.: Temporal phenotyping from longitudinal electronic health records: A graph based framework. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015. pp. 705–714 (2015), http://doi.acm.org/10.1145/2783258.2783352
9. Liu, C., Zhang, K., Xiong, H., Jiang, G., Yang, Q.: Temporal skeletonization on sequential data: Patterns, categorization, and visualization. IEEE Trans. Knowl. Data Eng. 28(1), 211–223 (2016), http://dx.doi.org/10.1109/TKDE.2015.2468715
10. McGovern, A., Rosendahl, D.H., Brown, R.A., Droegemeier, K.: Identifying predictive multi-dimensional time series motifs: an application to severe weather prediction. DMKD 22(1-2), 232–258 (2011), http://dx.doi.org/10.1007/s10618-010-0193-7
11. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. CoRR abs/1301.3781 (2013), http://arxiv.org/abs/1301.3781
12. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013 (NIPS 2013). Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States. pp. 3111–3119 (2013), http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and/-their-compositionality
13. Pathak, J., Kho, A.N., Denny, J.C.: Electronic Health Records-driven Phenotyping: Challenges, Recent Advances, and Perspectives. AMIA 20(December) (2013)
14. Pei, J., Dong, G., Zou, W., Han, J.: On computing condensed frequent pattern bases. In: Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM 2002), 9-12 December 2002, Maebashi City, Japan. pp. 378–385 (2002), http://dx.doi.org/10.1109/ICDM.2002.1183928
15. Pei, J., Han, J., Mortazavi-Asl, B., Pinto, H., Chen, Q., Dayal, U., Hsu, M.: Prefixspan: Mining sequential patterns by prefix-projected growth. In: Proceedings of the 17th International Conference on Data Engineering (ICDE 2001), April 2-6, 2001, Heidelberg, Germany. pp. 215–224 (2001), http://dx.doi.org/10.1109/ICDE.2001.914830
16. Rand, W.M.: Objective criteria for the evaluation of clustering methods. Journal of the American Statistical Association 66(336), 846–850 (1971), http://www.jstor.org/stable/2284239
17. Tang, L., Cui, B., Li, H., Miao, G., Yang, D., Zhou, X.: Effective variation management for pseudo periodical streams. In: Proceedings of the ACM SIGMOD International Conference

on Management of Data, Beijing, China, June 12-14, 2007. pp. 257–268 (2007), http://doi.acm.org/10.1145/1247480.1247511
18. Wang, F., Lee, N., Hu, J., Sun, J., Ebadollahi, S., Laine, A.F.: A framework for mining signatures from event sequences and its applications in healthcare data. IEEE Trans. Pattern Anal. Mach. Intell. 35(2), 272–285 (2013), http://dx.doi.org/10.1109/TPAMI.2012.111
19. Wang, P., Wang, H., Wang, W.: Finding semantics in time series. In: Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2011, Athens, Greece, June 12-16, 2011. pp. 385–396 (2011), http://doi.acm.org/10.1145/1989323.1989364
20. Xin, D., Han, J., Yan, X., Cheng, H.: Mining compressed frequent-pattern sets. In: Proceedings of the 31st International Conference on Very Large Data Bases (VLDB 2005), Trondheim, Norway, August 30 - September 2, 2005. pp. 709–720 (2005), http://www.vldb2005.org/program/paper/thu/p709-xin.pdf
21. Ye, L., Keogh, E.J.: Time series shapelets: a novel technique that allows accurate, interpretable and fast classification. DMKD 22(1-2), 149–182 (2011), http://dx.doi.org/10.1007/s10618-010-0179-5
22. Zhang, K., Wang, Q., Chen, Z., Marsic, I., Kumar, V., Jiang, G., Zhang, J.: From categorical to numerical: Multiple transitive distance learning and embedding pp. 46–54 (2015), http://dx.doi.org/10.1137/1.9781611974010.6