IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 19, 2020, accepted March 23, 2020, date of publication March 30, 2020, date of current version April 16, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2983987

Anomaly Detection by Learning Dynamics From a Graph

JAEKOO LEE[1], (Member, IEEE), HO BAE[2], AND SUNGROH YOON[2,3], (Senior Member, IEEE)

[1]College of Computer Science, Kookmin University, Seoul 02707, South Korea
[2]Interdisciplinary Program in Bioinformatics, Seoul National University, Seoul 08826, South Korea
[3]Department of Electrical Engineering and Computer Science, Seoul National University, Seoul 08826, South Korea

Corresponding author: Sungroh Yoon (sryoon@snu.ac.kr)

This work was supported in part by the National Research Foundation of Korea (NRF) funded by the Korea Government under Grant NRF-2018R1C1B5086441 and Grant 2018R1A2B3001628.

ABSTRACT There exist relations, which vary with time or by an event, between high dimensional elements in most real-world datasets. A dynamic graph or network has been used as one of the remarkable approaches to represent and analyze them. In spite of the advantages of representing data in the form of graphs, it is difficult to apply representation (deep) learning to graphs. Recently, AlphaFold by DeepMind has shown remarkable results in applying deep learning to graphs. This research is part of the current effort to extend the input domain of deep learning to arbitrarily graphs and their dynamics of variations. In this paper, we propose a method to predict the evolution of graphs by learning spatio-temporal features called dynamics. The method involves two main processes: extracting spatial features from static graphs obtained at different times and learning temporal features from the time-varying connection structure. Instead of predicting the overall changes of a highly complex graph, we detect the dynamic anomaly by predicting the affinity score with respect to a node (e.g., a hub as an important factor) of a dynamics graph. This facilitates the learning dynamics of graphs having sparsity of connections by alleviating the curse of dimensions using the fact that most graphs of real-world problems are scale-free. To justify our approach, we apply our method to real-world problems such as computer networks and public transportation. Experimental results show that our approach is competitive with other existing methods.

INDEX TERMS Deep learning, artificial neural network, anomaly detection, network (graph) theory, dynamic graph, spatial-temporal feature, affinity score, graph embedding, graph similarity.

I. INTRODUCTION

A graph is a powerful tool for solving complex tasks such as road traffic, computer networks, social network services, generating small molecular graphs, and even protein-protein interactions in bioinformatics [1]–[6].

Structurally, a data entity is represented by a node, and the relation between a pair of entities is represented by a weighted edge that connects the pair. A graph is a more flexible data representation than low-dimensional data structures in Euclidean space [2]. It provides greater adaptability in data abstraction. It could enable the development of new graph-based methods for understanding, visualizing, and analyzing data.

Traditionally, the analysis of data expressed in a graph has heavily relied on algebra, e.g., random walk [4], [7]–[10] and matrix completion [11], to capture the structural complexity and features, but machine learning has become increasingly popular as an alternative.

Deep learning has been successfully applied to data possibly represented in (low-dimensional) Euclidean space or grid domain, e.g., images, speech, and text [12], [13]. The domain of data to which deep learning is applicable has expanded in recent years even to non-Euclidean spaces such as a graph [14]–[17]. The method presented in this paper is intended for graph analysis and is a type of machine learning approach. The proposed deep learning model takes a graph that dynamically transforms over time or events as input and is trained using real-world datasets. One of the notable characteristics of a dynamic graph is that the properties change gradually rather than abruptly [10], as shown in Figure 1, which is involved in road networks. This observation contributed significantly to the motivation of our research.

The associate editor coordinating the review of this manuscript and approving it for publication was Chi-Yuan Chen.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

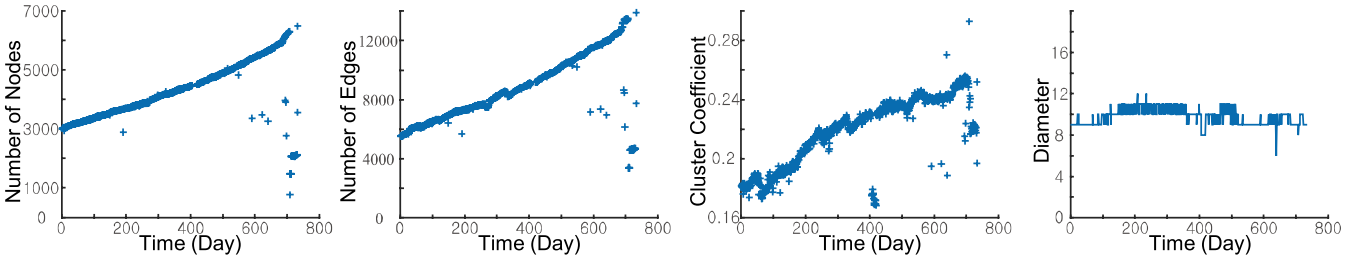


FIGURE 1. Visualization of the general properties of a real-world dynamic graph over time.

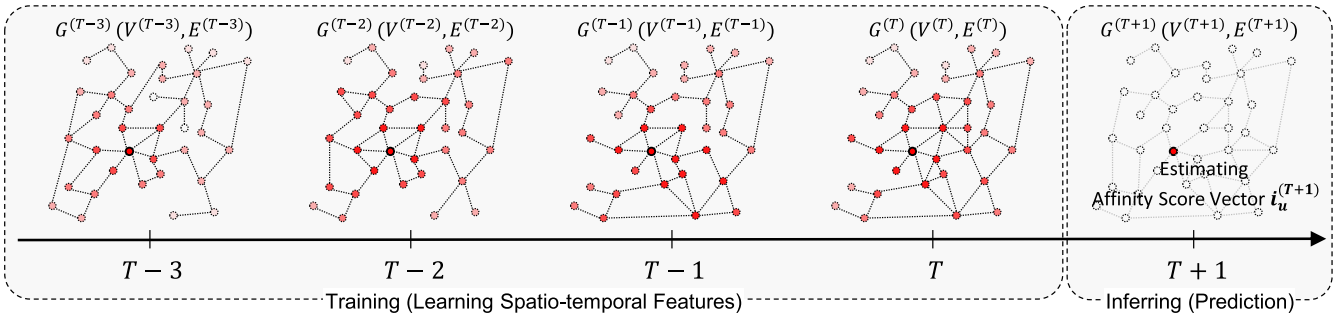


FIGURE 2. Description of the problem; training and inference of dynamic network transition in the aspect of the influence of a specific node (best viewed in color).

Well-known deep learning models, namely, convolutional neural networks (CNNs) [18] and recurrent neural networks (RNNs) [19], extract data-driven features in learning tasks where the coordinates of the underlying data representation have a grid structure; incidentally, both CNNs and RNNs hold statistical assumptions that an input data exhibit locality and stationary [15]. Unfortunately, these assumptions become invalid for dynamic graphs because a graph is defined in higher-dimensional non-Euclidean space. Learning on a graph needs to be redefined to extract features while preserving the structural features of a high-dimensional non-Euclidean domain.

The example in Figure 2 illustrates an overarching key aspect of our method. Specifically, it shows how we defined the task of learning a dynamically transforming graph by extracting data-driven spatio-temporal features. In our approach, a dynamic graph is viewed as a sequence of static graphs; the spatial features of each static graph are extracted while preserving the intrinsic geometric information of each static graph. After that, each extracted spatial feature is streamed to learn the temporal features of a dynamic graph. As a result, the learned model can be used to make a prediction about a subsequent graph.

When the training data set is a highly complex graph with a sparse connection, the curse of dimensionality [20] could become an issue. To alleviate this issue, the proposed method moderates it by simplifying the graph prediction problem by employing affinity score [4], [7]–[9]. Especially, the result is viewed from a specific node, enabling the quantitative prediction of changes in the entire graph. The reasons for this are as follows.

Most graphs and networks found in nature are scale-free: a hub node connects many nodes and provides important connections [1], [21]. In a dynamic graph that has scale-free graph characteristics, the prediction of affinity score vector could be used to quantify the overall changes of the graph indirectly from the perspective of an important node, e.g., a hub. Since an affinity score quantifies the structure of graph with respect to a specific node, it has been used as a major node proximity indicator in graph analysis such as cross-graph analytics, graph classification, link perdition, and anomaly detection in graphs [4], [7]–[10].

To verify the validity of our method and explore the training tendency of its model, we performed several experiments on the real-world dynamic graph datasets of public transportation and computer network traffic, both of which are available in public. The experimental results show that our deep neural networks is capable of extracting data-driven spatio-temporal features from a dynamic graph and that it can appropriately predict structural changes in the graph.

The results also confirm that the proposed method is applicable to the analysis of evolvable data with complex structures or interactive elements, e.g., intrusion (anomaly) detection [22] or biological metabolism [1], which involve complex interconnections between multiple constituents with dynamic behavior.

The contributions of our work can be summarized as follows:

- We established a new method for learning data-driven spatio-temporal features and predicting the change of dynamic connections in a graph, which transforms with time or by events.
- We validated, optimized, and utilized the proposed method based on experimental results using publically available real-world datasets.
- We proposed a novel approach to bridge the gap between deep learning methods and applications of graphs.

II. BACKGROUND

A. LEARNING ON GRAPH

The graph representation is a powerful tool since it enables the intuitive expression of highly complex structures that have interactive elements [2]–[4], [6], [10]. Graph analysis has seen significant development as deep learning has become more sophisticated. Notable topics include translation of data defined in a higher-dimensional non-Euclidean domain to lower-dimensional space.

Convolutional networks on graphs, for example, redefine generalized convolution in spectral domain [15]. Generalized convolution extracts data-driven features from graph-structured data. Graph convolutional networks [15], [17] replace the convolution operation of the existing neural network with the graph convolution (Laplacian) operation and extract the structural features of the graph by updating the weights of the neural network through gradient descent.

In the same context, some research formalized the hierarchical structure of graph connections as locality [16]. Transfer learning approaches for the graph domain make deep learning efficient for graph-structured data [17]. Graph embedding is traditionally an important issue. Recently, various embedding works have also been proposed on graphs [23], [24].

These studies have expanded the territory of deep learning into the graph domain by providing a more sophisticated training method. Some of the latest works extract data-driven features from graphs by means of deep learning to produce new graph representations [6].

B. ANOMALY DETECTION ON GRAPH

Anomaly detection has been widely applied in various applications such as statistics and machine learning. The most common approach for anomaly detection is to calculate the distance from each observed data to the normal dataset. The Euclidean distance is one popular measure for the data represented in low-dimensional space. One drawback of the Euclidean distance is that the approach does not consider the variance of each variable. The Mahalanobis distance uses a covariance matrix, which effectively represents the relationship between variables. The approach does not compute the distance from the mean value but uses its standard deviation and covariance of each variable [25]. Alternatively, statistical distribution can be used to calculate the distance such as Hotelling's T2 statistics [26]–[29].

For the non-Euclidean space, the distance can be measured by computing various measures that are associated with a node. The node-level features include followings: affinity score [4], [10]; direction of nodes (in/out); centrality measures; eigenvector [30]; closeness [31]; local clustering coefficient [32]; radius [33], degree assortativity; and roles [34]. With various types of distance measurement, a threshold value is established to detect anomaly behavior. An observed data is classified as an anomaly if the calculated distance of the feature is larger than the threshold.

C. RANDOM WALK-BASED GRAPH ANALYSIS

The graph, which is the main element of this research, is a domain and a tool where relationships and definitions can be represented. Its comprehensive researches were conducted in various fields and applications [35].

The random walk-based analysis is one of the well-known graph analysis approaches [4], [7]–[10], [36]. In this approach, a particle explores the input graph by "walking" from one node to another depending on the connection state between the nodes. During exploration, the structural characteristics of the graph in the high-dimensional non-Euclidean domain are quantified while preserving its intrinsic structure. Similar to random walk-based analysis, message-passing algorithms such as belief propagation can analyze structural features progressively disseminating evidences of the relativity information between nodes via edge connections and its weights. But, these algorithms can struggle when the graph contains loops or mixed connections of direct and undirect [37].

These approaches have been applied for graph structure analysis and measurement of the similarity between graphs by quantifying the interactive attraction or affinity score [4], [10]. Notable examples of the approach include PageRank [9], electrical network analogy [36], and personalized random walk with restart [7].

We applied a state of the art random walk-based method to obtain the affinity score, which depends on the structural characteristics of the input graph.

D. SPATIO-TEMPORAL LEARNING

CNNs-based methods have been used for learning spatial features from grid domain (e.g., images) [18], while RNNs-based methods have been useful for learning temporal features from sequential data such as text [19] and speech [38]–[40].

However, activity recognition in robotics and object recognition in videos broadly belong to both spatial domain and temporal domain simultaneously; thus, such problems can be addressed only through integrated approaches [41]. Many other interesting applications exist for spatio-temporal learning in rich interaction [42], robotics [43], autonomous vehicles [44], and graph [45]. The spatio-temporal learning method proposed in this paper is applicable to data that can be expressed properly using a graph.

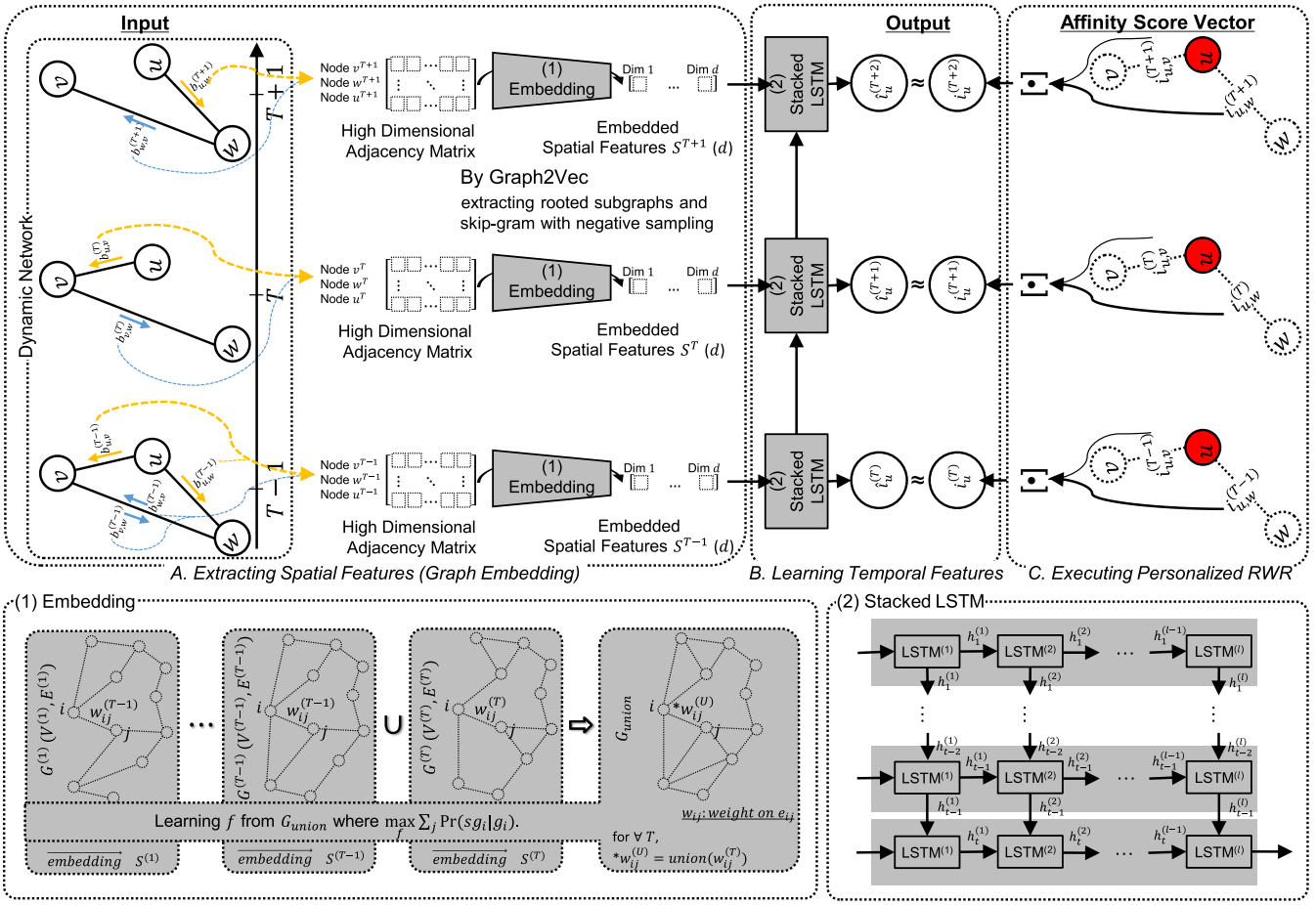


FIGURE 3. **[TOP] Overview of the proposed method (best viewed in color). [BOTTOM] Detailed description of the main parts of the method. (1) Embedding (2) Stacked LSTM.**

III. PROBLEM STATEMENT

Given: A dynamic graph $\mathbf{G}(\mathbf{V}, \mathbf{E}) = \{G^{(1)}(V^{(1)}, E^{(1)}), G^{(2)}(V^{(2)}, E^{(2)}), \ldots, G^{(T)}(V^{(T)}, E^{(T)})\}$ is defined as a sequence of static graphs up to a specific point of time $t = T$. $V^{(t)}$ is a set of nodes. To simplify the complexity of the graph, it is assumed that the node correspondence ($V^{(1)} = V^{(2)} = \cdots = V^{(T)}$, and $|V| = n$) of a time-varying graph is known [2], [4], [10]. A set of edges is represented by $E^{(t)} = e_{u,v}^{(t)}$; $u, v \in V^{(t)}$, and it is assumed that there is no self-loop, yet the connection state of edge $E^{(t)}$ can be changed with respect to t.

Find: We predict the change of graph $G^{(T+1)}(V^{(T+1)}, E^{(T+1)})$ at $t = T + 1$ by learning the spatio-temporal features from a given $\mathbf{G}(\mathbf{V}, \mathbf{E})$. The change of $G^{(T+1)}(V^{(T+1)}, E^{(T+1)})$ is estimated using the affinity score vector $i_u^{(T+1)} = [i_{u,1}^{(T+1)}, i_{u,2}^{(T+1)}, \ldots, i_{u,n}^{(T+1)}]$, which quantifies the structural relevance between a specific node u and another node w ($w \neq u$). We assume that the affinity score from a special node u (e.g., a hub) indirectly represents the entire graph using scale-free property. Predicting the affinity score is also an important subject as shown in previous works for graphs [4], [7]–[10]. Furthermore, it helps reduce the complexity of data representations and facilitate the learning of graphs.

IV. METHODS

The proposed method is illustrated conceptually in Figure 3 and consists of two parts. Under considering a given dynamic graph to be a set of consecutive static graphs, the spatial features from each static graph are extracted in advance. Next, the temporal features are learned according to the continuity of the spatial features. An integrated reasoning approach becomes practical in aspects of the spatial domain of the dynamic graph and its corresponding temporal domain. Consequently, the dynamics of a graph can be predicted from data-driven spatio-temporal feature learning.

A. EXTRACTING SPATIAL FEATURES

Since a static graph has a high-dimensional non-Euclidean domain, it is difficult for existing deep learning methods such as CNNs and RNNs, which take relatively low-dimensional vectors or matrices as grid input, to learn directly the structure of a graph [15], [17]. There have been approaches and attempts to solve this problem in applications having the high-dimensional domain, including graphs. Natural language processing showed remarkable results for the problem.

To unravel high-dimensional domain problems related to natural language processing, techniques producing the embedded vectors from words are utilized [46], [47]. One of the representative algorithms for Word2Vec (Doc2Vec) is a skip-gram modeling method that generates the low-dimensional embedded vector corresponding to a high-dimensional word based on the mutual occurrence frequencies of words (for detailed definitions, see [47]). The mutual occurrence frequency indicates the frequency with which certain sequences of words occur under a sliding window. Similarly, there are algorithm for embedding a graph [15], [24], [48].

In our method, embedding for a static graph maps a graph region to a low-dimensional region while preserving the adjacency and substructure of the nodes. To achieve proper embedding results for our task, we adopt the Graph2Vec algorithm, which is a data-driven representation learning approach, by comparing various graph embedding algorithms. Graph2Vec is inspired by Doc2Vec, which exploits how sequences of words compose documents to learn their embedding representations. Graph2Vec learns their representations from a corpus of graphs by considering the set of all rooted subgraphs (i.e., neighborhoods) around every node as its vocabulary.

The algorithm consists of two main components; first, a procedure to generate rooted subgraphs around every node in a given static graph and second, the procedure to learn embeddings of the given graphs. Subsequently, following training process of the Doc2Vec skip-gram with negative sampling [24], [47]), we learn the embedded representations of each graph from a given set $G_{union}(V, E) = \{g_1, g_2, \ldots, g_N\}$ of static graphs and a sequence of rooted subgraphs $c(g_i) = \{sg_1, sg_2, \ldots, sg_{l_i}\}$ sampled from graph $g_i \in G_{union}$.

$$\max \sum_j \log Pr(sg_j|g_i) \tag{1}$$

where the probability $Pr(sg_j|g)$ is defined as $\frac{\exp(g \cdot sg_j)}{\sum_{sg} \exp(g \cdot sg_j))}$ by independent assumption $\prod_j Pr(sg_j|g)$.

The algorithm makes embedded vectors able to preserve the structural characteristics of the graph presented in high-dimensional non-Euclidean domain. The embedded representation is learned by graph embedding in each static graph $G^{(t)}(N^{(t)}, E^{(t)})$ $(1 \leq t \leq T)$ to acquire spatial features $(S^{(t)} \in \mathbb{R}^d)$, respectively. Intuitively, $Graph2Vec(G^{(t)}(N^{(t)}, E^{(t)})) \rightarrow \mathbb{R}^d$ is performed to map the graph obtained at each time to its feature space in the form of a d-dimensional vector.

We assume a node correspondence in here, but the size of the node is not fixed. This is because we set enough nodes of the union graph, and we could vary the number of nodes variably. Most dynamic graphs or problems dealing with multiple graphs deal with the variable node size of the graph through this assumption. Therefore, the nodes and edges of the input graph of our method are variable.

B. LEARNING TEMPORAL FEATURES

To predict the affinity score vector of a specific node, the spatial features $S^{(t)}$ extracted from the static graph at each point are applied to a stacked long short-term memory (stacked LSTM) model [49], [50] with its regularizer and optimizer [51] where the temporal features of the dynamic graph are supposed to be learned.

The affinity score vector is predicted as follows: Let $\mathbf{S} = \{S^{(1)}, S^{(2)}, \ldots, S^{(T)}\}$, where $\forall S \in \mathbb{R}^{|G_{union}| \times d}$, be a set of T consecutive projected spatial features. By modeling $\hat{i}_{t+1} = \arg\max_{i_{t+1}} Pr(i_{t+1}|S^{(t-k+1)}, S^{(t-k+2)}, \ldots, S^{(t)})$ for a given k observed S, the real affinity score vector i computed by a personalized random walk is given as the ground truth for the stacked LSTM. Based on the learning process, the change of the dynamic graph at the next time is estimated quantitatively.

Specifically, the x_t is learned by a model with multiple layers of LSTM defined as follows.

$$in_t = \sigma(W_{ix}x_t + W_{ih}h_{t-1} + W_{ic}c_{t-1} + b_i) \tag{2}$$
$$f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1} + W_{fc}c_{t-1} + b_f) \tag{3}$$
$$out_t = \sigma(W_{ox}x_t + W_{oh}h_{t-1} + W_{oc}c_t + b_o) \tag{4}$$
$$c_t = f_t \odot c_{t-1} + in_t \odot \tanh(W_{cx}x_t + W_{ch}h_{t-1} + b_c) \tag{5}$$
$$h_t = out_t \odot \tanh(c_t) \tag{6}$$

where c_t: memory cell, h_t: hidden state, in_t: input gate, f_t: forget gate, out_t: output gate, and $\odot$: element-wise multiplication [49], [50].

The extension from LSTM to a stacked LSTM is defined as $h_t^{(l)} = f_h^{(l)}(h_t^{(l-1)}, h_{t-1}^{(l)}) = \sigma(W_l h_{t-1}^{(l)} + U_l h_t^{(l-1)})$ where $h_{t^{(l)}}$ is the hidden state of the l-th level at t and $h_t^{(0)} = x_t$. Stacked LSTM models are learned using well-known truncated back-propagation through time method [52].

The loss function is $L = \sum_n (\hat{i} - i)^2$ where i and $\hat{i}$ are the actual affinity score vector of the graph at the next time and the vector predicted by the model respectively. The model is trained by minibatch, weight-dropped LSTM with the averaged stochastic gradient optimizer (AWD-LSTM) [49], [51].

C. EXECUTING PERSONALIZED RWR

Due to the high complexity of the graph and the curse of dimensionality, we employed the spatio-temporal features learning approach to estimate the dynamics of an entire graph by predicting the affinity score vector i. In other words, after substituting the ground truth of the prediction with the affinity score vector by taking advantage of the characteristic of a scale-free graph with the sparsity of connections, the learning could be performed effectively.

The affinity score $i_{u,w}$ is a vector created by quantifying how many nodes u affects node w depending on the edge structure. Node u has more influence ($i_{u,w}$ is larger) on node w if the paths from node u to node w are numerous, short, or heavily weighted [4], [7]–[10]. $i_{u,w}$ lies in an interval $[0, 1]$, and $\sum_w i_{u,w} = 1$. Therefore, i is a vector having stochastic quantified elements of a structural affinity between u node and the other. To estimate the change of the dynamic

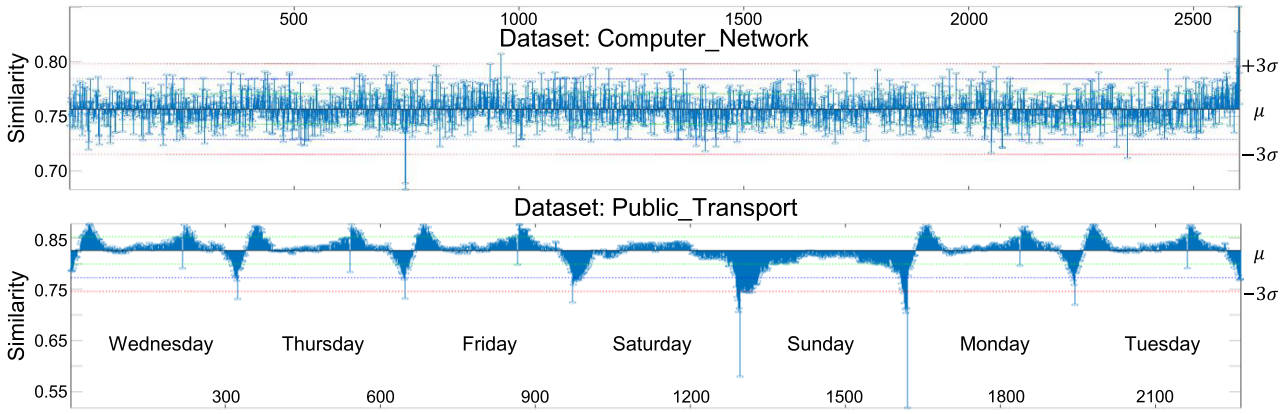


FIGURE 4. Visualization of consecutive graphs of used datasets in terms of dynamics.

TABLE 1. Used real-world dynamic graph (network) datasets.

Name	Nodes	Duration (Interval)	Description
Public_Transport (PT)	580	7 days (5min)	Movement flow of subway passengers in a large city
Computer_Network (CN)	225	9 days (5min)	Simulated cyber challenge network traffic flow

graph accurately, the affinity score vector should be attained efficiently and precisely.

The affinity score vector was calculated using a variation of a personalized random walk with a restart (RWR) on each static graph. The RWR is an algorithm that obtains an affinity score that quantifies the graph linkage structure viewed from a specific node through matrix operations in existing graph studies. Therefore, applying the RWR algorithm to the nodes at the center of the graph connection like the hub simply quantifies the dynamics of the graph to attenuate the cures of dimensionality. The specific node can be selected not only for the hub but also for other nodes, and generalization is possible because it can be learned and predicted through the corresponding node. It can be expanded even with multiple nodes of interest, such as when multiple hubs exist.

For a given graph $G(V, E)$ at a particular t, the corresponding adjacency matrix $\mathbf{A}$ consists of connection weights $a_{i,j}$ of node i and node j in the form of an $n \times n$ matrix, and the diagonal degree matrix $\mathbf{D}$ is defined as $d_{i,i} = \sum_j a_{ij}$. The affinity or influence vector $i_u \in \mathbb{R}^n$ for a specific node u from personalized random walk with restart is defined as follows:

$$i_u^{(k)} \leftarrow (1-c)\mathbf{A}\mathbf{D}^{-1} i_u^{(k-1)} + c\vec{e}_u \tag{7}$$

where $c \in [0, 1]$ is the probability of restarting the random walk from the initial node, $\vec{e}_u$ is the starting indicator vector (all zeros excepts 1 at the u-th element), and (k) is the number of iterations of random walk. As a result, it can be defined as an asymptotic equation

$$[\mathbf{I} - (1-c)\mathbf{A}\mathbf{D}^{-1}]i_u = ce_u \tag{8}$$

where $\mathbf{I}$ is an identity matrix.

The applied asymptotic equations have time complexity less than the initial approach (Eq. 7) of obtaining affinity score by initialization and iteration [4], [10]. In our method, the affinity score i_u, obtained from the Eq. 8, is the ground truth of supervised learning for predicting $\hat{i}_u$. The prediction of $\hat{i}_u$ extends to detect anomaly in structural changes of a dynamic graph. To predict the structural anomaly behaviors of the dynamic graph, we first calculate the differentiation (similarity) of affinity scores between successive static graphs to estimate their distributions, as shown in Figure 4. Following previous studies [4], [10], we define the label (the structural change of the graph) as an anomaly if the differentiation (similarity) distributions fall outside $\pm\sigma$ or $\pm 2\sigma$ otherwise define as normal.

The proposed neural network can be fine-tuned to predict the structural anomaly of the dynamic graph is determined by the i_u as a classification task by using transfer learning from the $\hat{i}_u$ prediction task.

V. EXPERIMENTS AND DISCUSSION

We evaluated the proposed method using real-world dynamic graph datasets, such as network traffic and public transportation traffic. The details of datasets are shown in Table 1. Both data have already been used in previous works on data analysis and are thought to be sufficiently representative of data in the computer and traffic network sectors.

In the used datasets, a few nodes (hubs) are characterized as highly connected to other nodes in the network. We, therefore, use the affinity score of the hub at the center of the connection to learn and predict the dynamics of the network change efficiently. In Figure 4, the change of similarity between

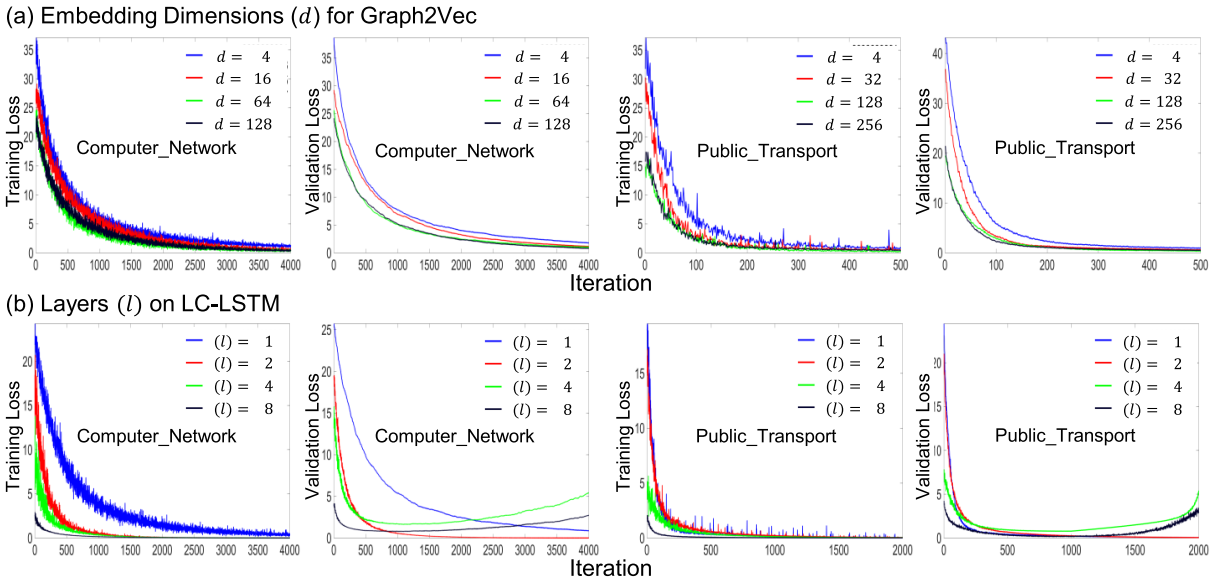


FIGURE 5. Training and validation loss (best viewed in color) results for (a) various ds with basic LSTM ($(l) = 1$) and (b) various (l)s in the stacked LSTM.

consecutive graphs on datasets is plotted. $sim(G_1, G_2) = 0$ indicates that two graphs G_1 and G_2 are structurally complementary (absolutely different graphs), whereas a value of 1 means that they are identical. From the visualization of dynamics on a graph, we quantitatively confirmed how the graph changes.

To verify the performance of hyper-parameters on models, the training tendency for an optimal model was examined by the loss function. Each dataset was divided into training data (80%) and validation data (20%). A specific node u was chosen among the hub nodes based on the knowledge from previous works [3], [10]. We checked the loss trends for predicting i_u and compared its performance to optimize the hyper-parameters.

The experimental results for meaningful hyper-parameters are as follows. To confirm the *effects of embedding* on the spatial feature learning, we observed the changes in the training curves that were caused by adjusting the embedding dimension (d). As shown in Figure 5 (a), regardless of d, the training and validation losses decrease as the number of iterations increases. We also confirmed that there is a tiny gap in performance according to an increase of d. These decreases in loss are regarded as evidence of proper training. From the result, an appropriate d can be chosen when d is about $|V|$. Based on these facts, the convergence of training can be achieved properly.

We also conducted experiments on stacked LSTM with *capacity and complexity of model* by gradually adjusting the number of layers (l). Figure 5 (b) exhibits an appropriate capacity of the model without overfitting. These experiments yielded that the optimal model is stacked LSTM$^{(2)}$.

To confirm the correlation between measurements for a change of a dynamic graph, additional experiments were conducted. Figure 6 shows a strong correlation between the following three measurements. Entire (E): quantified change in an entire dynamic graph using a similarity measurement, as shown in Figure 4. Hub (H): quantified change in real affinity scores, which is represented in a vector, for the hub. Prediction (P): quantified change in an affinity score estimated by the proposed method. From the results in Figure 6, the dynamics of the entire graph and the dynamics in real affinity score on the hub are highly correlated as well as the change in real and predicted affinity score is highly correlated. Furthermore, the anomalous signs confirmed in the previous studies [3], [10] were also identified by our method. It shows that our method is able not only to predict anomalous signs but also to analyze similarity in the graph.

We also compare the results of the optimal model obtained from the experiments against other up-to-date techniques, namely RgCNN (recurrent graph convolutional neural networks) [54], DGCN (dynamic graph convolutional networks) [55], CTRBM (conditional temporal restricted Boltzmann machine) [53], and SEAL (learning from subgraphs, embedding, and attribute for link prediction) [56] which are state-of-the-art learning algorithms for graph prediction.

Table 2 shows a comparison between techniques in terms of the sum of absolute differences (*SumD*) for the prediction of the affinity score of the hub and accuracy (*AUC*; the area under a receiver operating characteristic curve) for the prediction of existing links. As can be seen from the detailed results in the table 2, our method shows relatively competitive performance. The result with low *SumD* and high *AUC* indicates better performance of predicting graph dynamics. This result implies that ours is better than other techniques in terms of the performance of link prediction considering a weighted edge.

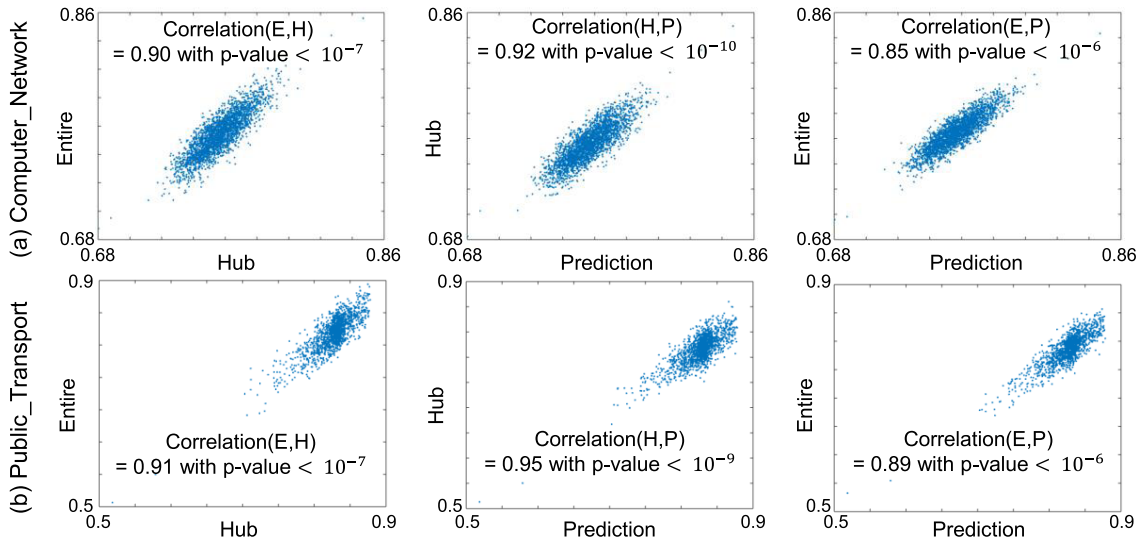


FIGURE 6. Correlation between variations in measuring graph similarity.

TABLE 2. Overall comparison with other techniques.

Performance	*SumD*		*AUC*	
Dataset	Computer_Network	Public_Transport	Computer_Network	Public_Transport
CTRBM [Li et al.2014]	224.3532	43.3212	0.6102	0.7016
RgCNN [Narayan and Roe2018]	183.4924	17.5094	0.6906	0.8610
DGCN [Manessi et al.2017]	174.0042	13.4032	0.7021	0.8705
SEAL [Zhang and Chen2018]	171.9832	12.0298	0.7201	0.9105
OURS	**148.2091**	**10.2941**	**0.7202**	**0.9224**

TABLE 3. Comparative evaluation of anomaly detection on graph similarity threshold.

Anomaly detection	*AUC*			
Dataset	CN		PT	
Outside threshold	$\pm 1\sigma$	$\pm 2\sigma$	$\pm 1\sigma$	$\pm 2\sigma$
CTRBM [Li et al.2014]	0.58	0.54	0.69	0.68
RgCNN [Narayan and Roe2018]	0.61	0.60	0.76	0.78
DGCN [Manessi et al.2017]	0.63	0.60	0.75	0.80
SEAL [Zhang and Chen2018]	0.71	0.70	0.82	0.80
OURS	**0.76**	**0.74**	**0.88**	**0.86**

As shown in the Table 2, we have observed that our approach outperformed than other recent studies in overall performance aspects. In addition, we have evaluated the prediction performance of our method using a conventional benchmark by measuring anomalies in a dynamic graph.

Both Table 3 and Figure 4 represent the result of predicting anomalies that are above the threshold (standard deviation) in a dynamic graph. In general, the public_transport data varies with periodic variations, whereas the computer_network data has a pattern of substantial changes. We noticed both from the Table 2 and Table 3 that our approach performed better in the public_transport data.

Our method also consider efficient learning in task changes. Prediction anomalies in a dynamic graph have been efficiently learned by transfer learning. In terms of efficiency, the model learns faster (21%) with higher asymptotic accuracy (3%) than those in the baseline case in which the neural network is trained from scratch. Similar to the findings from previous studies [17], the result confirms that the transfer learning not only efficiently learn a domain shift or a distribution change between the source and the target but also changes of a task in deep neural networks for graphs.

VI. CONCLUSION

Due to its ability to represent elements and their relations, a graph is a highly promising data representation. Nonetheless, the complexity of the non-Euclidean structure of a graph makes analysis difficult. Our work suggests a new dynamic graph analysis method using deep learning. The proposed method enables the application of deep learning to a dynamic graph, even though deep learning is mainly used in low-dimensional grid domains such as speech and images. We considered transfer learning for efficient learning in task changes. To demonstrate the effectiveness of learning data-driven spatio-temporal features from a dynamic graph, we conducted various experiments on real-world datasets

such as computer network and transportation traffic. Our method is also competitive with state-of-the-art techniques. We expect our proposed model to be employed in various dynamic graph applications, including anomaly detection.

REFERENCES

[1] A. Wagner and D. A. Fell, "The small world inside large metabolic networks," *Proc. Roy. Soc. London. Ser. B, Biol. Sci.*, vol. 268, no. 1478, pp. 1803–1810, Sep. 2001.
[2] C. C. Aggarwal and H. Wang, *Managing and Mining Graph Data*, vol. 40. New York, NY, USA: Springer, 2010.
[3] S. Rayana and L. Akoglu, "An ensemble approach for event detection and characterization in dynamic graphs," in *Proc. 2nd ACM SIGKDD Workshop Outlier Detection Description Under Data Diversity (ODD)*, 2014.
[4] D. Koutra, J. T. Vogelstein, and C. Faloutsos, "DeltaCon: A principled massive-graph similarity function," in *Proc. SIAM Int. Conf. Data Mining*. Philadelphia, PA, USA: SIAM, May 2013, pp. 162–170.
[5] N. De Cao and T. Kipf, "MolGAN: An implicit generative model for small molecular graphs," 2018, *arXiv:1805.11973*. [Online]. Available: http://arxiv.org/abs/1805.11973
[6] D. K. Duvenaud, D. Maclaurin, J. Iparraguirre, R. Bombarell, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams, "Convolutional networks on graphs for learning molecular fingerprints," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 2224–2232.
[7] T. H. Haveliwala, "Topic-sensitive PageRank," in *Proc. 11th Int. Conf. World Wide Web (WWW)*, 2002, pp. 517–526.
[8] J. S. Yedidia, W. T. Freeman, and Y. Weiss, "Understanding belief propagation and its generalizations," *Exploring Artif. Intell. New Millennium*, vol. 8, pp. 236–239, Jan. 2003.
[9] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank citation ranking: Bringing order to the Web," Stanford InfoLab, Stanford, CA, USA, Tech. Rep. 1999-0120, 1999.
[10] J. Lee, G. Kim, and S. Yoon, "Measuring large-scale dynamic graph similarity by RICom: RWR with intergraph compression," in *Proc. IEEE Int. Conf. Data Mining*, Nov. 2015, pp. 829–834.
[11] V. Kalofolias, X. Bresson, M. Bronstein, and P. Vandergheynst, "Matrix completion on graphs," 2014, *arXiv:1408.1717*. [Online]. Available: http://arxiv.org/abs/1408.1717
[12] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
[13] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.
[14] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," 2019, *arXiv:1901.00596*. [Online]. Available: http://arxiv.org/abs/1901.00596
[15] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," 2013, *arXiv:1312.6203*. [Online]. Available: http://arxiv.org/abs/1312.6203
[16] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3837–3845.
[17] J. Lee, H. Kim, J. Lee, and S. Yoon, "Transfer learning for deep learning on graph-structured data," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 2154–2160.
[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
[19] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3104–3112.
[20] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. Cambridge, MA, USA: MIT Press, 2012.
[21] A. Clauset, C. R. Shalizi, and M. E. J. Newman, "Power-law distributions in empirical data," *SIAM Rev.*, vol. 51, no. 4, pp. 661–703, Nov. 2009.
[22] M. E. Whitman and H. J. Mattord, *Principles of Information Security*. Boston, MA, USA: Cengage Learning, 2011.
[23] A. Grover and J. Leskovec, "Node2vec: Scalable feature learning for networks," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2016, pp. 855–864.
[24] A. Narayanan, M. Chandramohan, R. Venkatesan, L. Chen, Y. Liu, and S. Jaiswal, "Graph2vec: Learning distributed representations of graphs," 2017, *arXiv:1707.05005*. [Online]. Available: http://arxiv.org/abs/1707.05005
[25] D. Bayarjargal and G. Cho, "Detecting an anomalous traffic attack area based on entropy distribution and mahalanobis distance," *Int. J. Secur. Its Appl.*, vol. 8, no. 2, pp. 87–94, Mar. 2014.
[26] F. Harrou, Y. Sun, and S. Khadraoui, "Amalgamation of anomaly-detection indices for enhanced process monitoring," *J. Loss Prevention Process Industries*, vol. 40, pp. 365–377, Mar. 2016.
[27] H. Om and T. Hazra, "Statistical techniques in anomaly intrusion detection system," *Int. J. Adv. Eng. Technol.*, vol. 5, no. 1, p. 387, 2012.
[28] M.-L. Shyu, S.-C. Chen, K. Sarinnapakorn, and L. Chang, "A novel anomaly detection scheme based on principal component classifier," Dept. Elect. Comput. Eng., Univ. Miami, oral Gables, FL, USA, Tech. Rep. 2003-01-01, 2003.
[29] A. Lakhina, M. Crovella, and C. Diot, "Diagnosing network-wide traffic anomalies," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 4, p. 219, Oct. 2004.
[30] P. Bonacich and P. Lloyd, "Eigenvector-like measures of centrality for asymmetric relations," *Social Netw.*, vol. 23, no. 3, pp. 191–201, Jul. 2001.
[31] J. D. Noh and H. Rieger, "Random walks on complex networks," *Phys. Rev. Lett.*, vol. 92, no. 11, Mar. 2004, Art. no. 118701.
[32] D. J. Watts, "Networks, dynamics, and the small-world phenomenon," *Amer. J. Sociol.*, vol. 105, no. 2, pp. 493–527, 1999.
[33] U. Kang, C. E. Tsourakakis, A. P. Appel, C. Faloutsos, and J. Leskovec, "HADI: Mining radii of large graphs," *ACM Trans. Knowl. Discovery from Data*, vol. 5, no. 2, pp. 1–24, Feb. 2011.
[34] K. Henderson, B. Gallagher, T. Eliassi-Rad, H. Tong, S. Basu, L. Akoglu, D. Koutra, C. Faloutsos, and L. Li, "RolX: Structural role extraction & mining in large graphs," in *Proc. 18th ACM SIGKDD Int. Conf. Knowl. discovery data mining*, 2012, pp. 1231–1239.
[35] S. S. Sonawane and P. A. Kulkarni, "Graph based representation and analysis of text document: A survey of techniques," *Int. J. Comput. Appl.*, vol. 96, no. 19, pp. 1–8, 2014.
[36] P. G. Doyle and J. L. Snell, *Random Walks and Electric Networks*. Washington, DC, USA: Mathematical Association of America, 1984.
[37] K. Yoon, R. Liao, Y. Xiong, L. Zhang, E. Fetaya, R. Urtasun, R. Zemel, and X. Pitkow, "Inference in probabilistic graphical models by graph neural networks," 2018, *arXiv:1803.07710*. [Online]. Available: http://arxiv.org/abs/1803.07710
[38] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2013, pp. 6645–6649.
[39] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, "End-to-end attention-based large vocabulary speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2016, pp. 4945–4949.
[40] D. Amodei *et al.*, "Deep speech 2 : End-to-end speech recognition in English and Mandarin," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 173–182.
[41] A. Jain, A. R. Zamir, S. Savarese, and A. Saxena, "Structural-RNN: Deep learning on spatio-temporal graphs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 5308–5317.
[42] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, Jan. 2016.
[43] I. Lenz, H. Lee, and A. Saxena, "Deep learning for detecting robotic grasps," *Int. J. Robot. Res.*, vol. 34, nos. 4–5, pp. 705–724, Apr. 2015.
[44] A. Jain, H. S. Koppula, B. Raghavan, S. Soh, and A. Saxena, "Car that knows before you do: Anticipating maneuvers via learning temporal driving models," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 3182–3190.
[45] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph WaveNet for deep spatial-temporal graph modeling," 2019, *arXiv:1906.00121*. [Online]. Available: http://arxiv.org/abs/1906.00121
[46] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013, *arXiv:1301.3781*. [Online]. Available: http://arxiv.org/abs/1301.3781

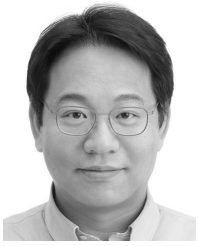
[47] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.
[48] P. Goyal and E. Ferrara, "Graph embedding techniques, applications, and performance: A survey," *Knowl.-Based Syst.*, vol. 151, pp. 78–94, Jul. 2018.
[49] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
[50] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio, "How to construct deep recurrent neural networks," 2013, *arXiv:1312.6026*. [Online]. Available: http://arxiv.org/abs/1312.6026
[51] S. Merity, N. Shirish Keskar, and R. Socher, "Regularizing and optimizing LSTM language models," 2017, *arXiv:1708.02182*. [Online]. Available: http://arxiv.org/abs/1708.02182
[52] I. Sutskever, "Training recurrent neural networks," Ph.D. dissertation, Dept. Comput. Sci., Univ. Toronto, Toronto, ON, Canada, 2013.
[53] X. Li, N. Du, H. Li, K. Li, J. Gao, and A. Zhang, "A deep learning approach to link prediction in dynamic networks," in *Proc. SIAM Int. Conf. Data Mining*. Philadelphia, PA, USA: SIAM, Apr. 2014, pp. 289–297.
[54] A. Narayan and P. H. O'N Roe, "Learning graph dynamics using deep neural networks," *IFAC-PapersOnLine*, vol. 51, no. 2, pp. 433–438, 2018.
[55] F. Manessi, A. Rozza, and M. Manzo, "Dynamic graph convolutional networks," 2017, *arXiv:1704.06199*. [Online]. Available: http://arxiv.org/abs/1704.06199
[56] M. Zhang and Y. Chen, "Link prediction based on graph neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 5165–5175.



JAEKOO LEE (Member, IEEE) received the Ph.D. degree in electrical and computer engineering from Seoul National University, South Korea, in 2018. From 2011 to 2013, he was with LG Electronics R&D Campus, Seoul, South Korea. He held a Data Scientist position at SK Telecom Corporation, in 2018. He is currently an Assistant Professor with the College of Computer Science, Kookmin University, South Korea. His research interests include artificial intelligence (AI), machine learning, and data science and its applications.



HO BAE received the B.Sc. degree in computer science and the M.Sc. degree in information security from University College London, U.K., in 2007 and 2009, respectively. He is currently pursuing the Ph.D. degree with the Interdisciplinary Program in Bioinformatics, Seoul National University, South Korea. He was with SAP Labs Korea, from 2012 to 2016. His research interests include artificial intelligence and bioinformatics.



SUNGROH YOON (Senior Member, IEEE) received the B.S. degree in electrical engineering from Seoul National University, South Korea, in 1996, and the M.S. and Ph.D. degrees in electrical engineering from Stanford University, CA, USA, in 2002 and 2006, respectively. He held a Research positions at Stanford University and Synopsys Inc., Mountain View, CA, USA. From 2006 to 2007, he was with Intel Corporation, Santa Clara, CA, USA. He was an Assistant Professor with the School of Electrical Engineering, Korea University, from 2007 to 2012. He is currently a Professor with the Department of Electrical and Computer Engineering, Seoul National University, South Korea. His research interests include machine learning and its applications to bioinformatics. He was a recipient of the 2013 IEEE/IEEK Joint Award for Young IT Engineers.