2020 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)

Distill2Vec: Dynamic Graph Representation Learning with Knowledge Distillation

Stefanos Antaris
KTH Royal Institute of Technology
Hive Streaming AB
Sweden
antaris@kth.se

Dimitrios Rafailidis
Maastricht University
Netherlands
dimitrios.rafailidis@maastrichtuniversity.nl

***Abstract*—Dynamic graph representation learning strategies are based on different neural architectures to capture the graph evolution over time. However, the underlying neural architectures require a large amount of parameters to train and suffer from high online inference latency, that is several model parameters have to be updated when new data arrive online. In this study we propose Distill2Vec, a knowledge distillation strategy to train a compact model with a low number of trainable parameters, so as to reduce the latency of online inference and maintain the model accuracy high. We design a distillation loss function based on Kullback-Leibler divergence to transfer the acquired knowledge from a teacher model trained on offline data, to a small-size student model for online data. Our experiments with publicly available datasets show the superiority of our proposed model over several state-of-the-art approaches with relative gains up to 5% in the link prediction task. In addition, we demonstrate the effectiveness of our knowledge distillation strategy, in terms of number of required parameters, where Distill2Vec achieves a compression ratio up to 7:100 when compared with baseline approaches. For reproduction purposes, our implementation is publicly available at https://stefanosantaris.github.io/Distill2Vec.**

***Index Terms*—Dynamic graph representation learning, knowledge distillation, model compression**

I. Introduction

Dynamic graph representation learning is a fundamental problem, with ubiquitous applicability in various real-world domains [1], [2]. To efficiently capture the evolution in the latent embedding space, dynamic graph representation learning approaches compute node embeddings based on a sequence of graph snapshots at different time steps [1]–[4]. Existing approaches explore several techniques to accurately learn node embeddings, such as temporal regularizers [2], [5], Recurrent Neural Networks [1], [6], and joint-self attention mechanisms [3].

Although dynamic graph representation learning strategies produce accurate predictions, they are based on deep neural network architectures with a large number of model parameters. Moreover, the number of parameters significantly increases by several orders of magnitude, along with the number of graph snapshots. Due to the vast amount of model parameters such approaches incur high online inference latency, which prohibits their direct applications into a real-world setting with almost real-time response requirements [7]–[11]. For example, the model size negatively impacts the performance of recommendation systems in social networks, where predictions have to be calculated in real time [11], [12].

Knowledge distillation is a model independent strategy to generate compact models that exhibit low online inference latency. [7], [8]. The basic idea of knowledge distillation is to train a large model, namely *teacher*, as an offline process. The teacher model can employ computationally expensive deep neural networks, as there are no strict requirements on latency and computational resources during offline learning. Having trained the teacher model, the knowledge can be transferred to a smaller model, namely *student*, by reducing the model size. Therefore, the student model can be deployed to online applications, satisfying the low online inference latency requirements [10], [11], [13]. However, the impact of knowledge distillation on graph representation learning for dynamic graphs has not been studied so far.

In this paper, we propose a knowledge distillation strategy, namely Distill2Vec, to generate a compact student model with low online inference latency for graph representation learning on dynamic graphs. The teacher model learns the latent node representations by employing a self-attention mechanism on the offline graph snapshots. To train a small student model on the online graph snapshots, we formulate a distillation loss function, allowing the student model to distill the knowledge of the pretrained teacher model. In doing so, the student model can generate similar predictions as the teacher model, while significantly reducing the model parameters. Our main contributions are summarized as follows: i) *We propose Distill2Vec, a knowledge distillation strategy on dynamic graph representation learning approaches. We formulate a distillation loss function based on Kullback-Leibler divergence to transfer the knowledge from the teacher model on the offline data, to a smaller student model when learning online data. In addition, Distill2Vec employs a self-attention mechanism to capture the graph evolution in the learned node embeddings.*; ii) *We demonstrate that the student model significantly reduces the online inference latency, in terms of the number of trainable parameters, when compared with the teacher model. Moreover, the proposed student model overcomes any bias introduced by the pretrained teacher model, achieving high accuracy in the online link prediction task.*

Our experiments on two real-world dynamic networks

IEEE/ACM ASONAM 2020, December 7-10, 2020
978-1-7281-1056-1/20/$31.00 © 2020 IEEE

demonstrate the superiority of our proposed knowledge distillation strategy, against several state-of-the-art methods.

The remainder of the paper is organized as follows: in Section II we describe the proposed knowledge distillation strategy. The experimental evaluation is presented in Section III and we conclude the study in Section IV.

II. Proposed Model

Dynamic graph representation learning models employ deep neural network architectures to learn accurate node embeddings, at the cost of high online inference latency [3], [5], [14]. The goal of our knowledge distillation strategy is to generate a compact student model $\mathcal{S}$ with low online inference latency, and retain the accuracy of the pretrained large teacher model $\mathcal{T}$ [7], [8], [15], [16]. In particular, the teacher model $\mathcal{T}$ is pretrained to learn the node embeddings $\mathbf{H}^{\mathcal{T}}$ on the offline graph snapshots. We denote by $\mathcal{G}^{\mathcal{T}} = \{\mathcal{G}_1, \ldots, \mathcal{G}_m\}$ the m consecutive offline graph snapshots of the dynamic graph $\mathcal{G}$, with $1 \leq m < T$. Thereafter, the student model $\mathcal{S}$ exploits the teacher model $\mathcal{T}$, to learn accurate node representations $\mathbf{H}^{\mathcal{S}}$ on the online graph snapshots $\mathcal{G}^{\mathcal{S}} = \{\mathcal{G}_{m+1}, \ldots, \mathcal{G}_T\}$. To transfer the knowledge of the pretrained teacher model $\mathcal{T}$, the student model $\mathcal{S}$ minimizes a distillation loss function $L^{\mathcal{D}}$ [7], [8]. The distillation loss function $L^{\mathcal{D}}$ calculates the prediction error of the student model $\mathcal{S}$ on the online graph snapshots, and the deviation of $\mathbf{H}^{\mathcal{S}}$ from the node embeddings $\mathbf{H}^{\mathcal{T}}$. In Section II-A, we describe the offline teacher model Distill2Vec-$\mathcal{T}$, and then in Section II-B we present the knowledge distillation strategy of the online student model Distill2Vec-$\mathcal{S}$.

A. Distill2Vec-$\mathcal{T}$ - Teacher Model

The teacher model Distill2Vec-$\mathcal{T}$ learns the latent representations $\mathbf{H}_t^{\mathcal{T}}$ based on the m offline graph snapshots $\mathcal{G}^{\mathcal{T}}$. Distill2Vec-$\mathcal{T}$ employs two self-attention layers [3], [17], [18]. The first layer, namely structural self-attention, captures the structural properties of each node $u \in \mathcal{V}_t$ at the t-th graph snapshot. The second layer, namely temporal self-attention, models the evolution of the graph, given a sequence of l graph snapshots $\{\mathcal{G}_{t-l}, \ldots, \mathcal{G}_t\}$. Provided that the teacher model Distill2Vec-$\mathcal{T}$ is trained as an offline process, we consider all the graph snapshots $\mathcal{G}^{\mathcal{T}}$ for the temporal self-attention layer ($l = m$). The input of the structural self-attention layer at the t-th time step is the set of input node representations $\mathbf{X}_t \in \mathbb{R}^{n_t \times n_t}$, where $\mathbf{X}_t(u)$ is the one-hot encoded vector of the node $u \in \mathcal{V}_t$. The output is a d-dimensional structural node representation $\mathbf{Z}_t(u) \in \mathbb{R}^d$, calculated as follows:

$$\mathbf{Z}_t(u) = ELU\Bigg(\sum_{u \in \mathcal{N}_t(u)} \alpha_t(u,v)\mathbf{W}_t\mathbf{X}_t(u)\Bigg) \tag{1}$$

where $\mathcal{N}_t(u)$ is the neighborhood set of the node u at the t-th time step, $\mathbf{W}_t \in \mathbb{R}^{d \times n_t}$ is the weight transformation matrix for each input node representation $\mathbf{X}_t(u)$, and ELU is the exponential linear unit activation function. Variable $\alpha_t(u,v)$ corresponds to the learned coefficients, calculated based on the softmax over the neighbors of each node u, as follows:

$$\alpha_t(u,v) = \frac{exp\big(e_t(u,v)\big)}{\sum\limits_{w \in \mathcal{N}_t(u)} exp\big(e_t(u,w)\big)} \tag{2}$$

$$\text{with} \quad e_t(u,v) = f\big(A_t(u,v) \cdot \mathbf{a}_t^{\top}[\mathbf{W}_t\mathbf{X}_t(u) \parallel \mathbf{W}_t\mathbf{X}_t(v)]\big)$$

f is the LeakyRelu activation function, $\mathbf{a}_t \in \mathbb{R}^{2n_t}$ is a $2n_t$-dimensional weight vector parameterizing the attention process between nodes u and v, and $\parallel$ denotes the concatenation operation. The attention weight $e_t(u,v)$ indicates the contribution of the node v to the node u at the t-th time step [3], [17].

Having computed the d-dimensional structural node representations $\mathbf{Z}_t$ for each time step $t = 1, \ldots, m$, we capture the graph evolution in the temporal attention layer. In contrast to the structural attention layer that learns the structural properties of the nodes at each time step, the temporal attention layer emphasizes on the evolution of each node over l consecutive graph snapshots, with $l = m$ for the teacher model. The input of the temporal attention layer, denoted by $\mathbf{X'}_t(u) \in \mathbb{R}^{l \times d}$, is calculated as $\mathbf{X'}_t(u) = Concat(\mathbf{Z}_{t-l}(u), \ldots, \mathbf{Z}_t(u))$, that is the concatenation of the l structural node representations of each node u. We apply the scaled dot-product form of attention [3], [18], where the structural node representations $\mathbf{X'}_t$ are the queries, keys and values of the attention process. For each node $u \in \mathcal{V}_t$, the temporal attention layer calculates l new k-dimensional representations $\mathbf{B}_t(u) \in \mathbb{R}^{l \times k}$ as follows:

$$\mathbf{B}_t(u) = \boldsymbol{\beta}_t(u)(\mathbf{X'}_t(u)\mathbf{W}_t^{value}) \tag{3}$$

$\mathbf{W}_t^{value} \in \mathbb{R}^{d \times k}$ is the linear projection matrix of the structural node representations of each node u. Variable $\boldsymbol{\beta}_t(u) \in \mathbb{R}^{l \times l}$ is the attention weight matrix that indicates the similarity of the node's u structural embeddings in different graph snapshots. For each graph snapshot $i = t-l, \ldots, t$ and $j = t-l, \ldots, t$, we calculate the attention weight of the node u as follows:

$$\beta^{ij}(u) = \frac{exp(c^{ij}(u))}{\sum\limits_{r=t-l}^{t} exp(c^{ir})(u)} \tag{4}$$

$$\text{with} \quad c^{ij}(u) = \left(\frac{((\mathbf{X'}_i(u)\mathbf{W}^{query})(\mathbf{X'}_j(u)\mathbf{W}^{key}))^{ij}}{\sqrt{k}} + M^{ij}\right)$$

$\mathbf{W}^{query} \in \mathbb{R}^{d \times k}$ and $\mathbf{W}^{key} \in \mathbb{R}^{d \times k}$ are the weight parameter matrices to transform the query and key input node representations, respectively [18]. A high attention weight $\beta^{ij}(u)$ corresponds to similar structural node embeddings for the node u in the graph snapshots $\mathcal{G}_i^{\mathcal{T}}$ and $\mathcal{G}_j^{\mathcal{T}}$. In Equation 4 $\mathbf{M} \in \mathbb{R}^{l \times l}$ is a mask matrix to encode the temporal order between different time steps i and j. The values of the matrix $\mathbf{M}$ are zero if $i < j$, and infinite otherwise.

We employ multi-head attention on both the structural and temporal attention layers, to capture the evolution of different latent facets over time for each node $u \in \mathcal{V}_t$ [3]. The output of the multi-head attention on the structural attention layer

is computed as $\mathbf{C}_t(u) = Concat(\mathbf{Z}_t^1(u), \ldots, \mathbf{Z}_t^h(u))$, where h is the number of attention heads and $\mathbf{C}_t(u) \in \mathbb{R}^d$ is the output representation of the node u at the t-th time step. Similar to the structural attention layer, the output of the multi-head attention on the temporal attention layer is defined as $\mathbf{D}_t(u) = Concat(\mathbf{B}^1(u), \ldots, \mathbf{B}_t^g(u))$, where g is the number of attention heads applied to the temporal attention layer and $\mathbf{B}_t(u) \in \mathbb{R}^{l \times k}$ is the output node representations of the node u.

Having computed both the structural and the temporal node representations, we can calculate the final node representation $\mathbf{H}_t(u)$ for each node $u \in \mathcal{V}_t$. We encode the ordering information in the node representations $\mathbf{D}_t(u)$ of the temporal attention layer, by calculating the position embeddings $\mathbf{P}_t(u) \in \mathbb{R}^d$ for each node u [19]. The final node representations $\mathbf{H}_t^{\mathcal{T}}(u)$ of the teacher model Distill2Vec-$\mathcal{T}$ are then computed by combining the output node representations $\mathbf{C}_t(u)$ of the structural attention layer with the position embeddings $\mathbf{P}_t(u)$ as follows:

$$\mathbf{H}_t^{\mathcal{T}}(u) = \mathbf{C}_t(u) + \mathbf{P}_t(u) \tag{5}$$

To train the teacher model and learn the node embeddings, we adopt the binary cross-entropy loss function with respect to the node embeddings $\mathbf{H}_t^{\mathcal{T}}(u)$:

$$\min_{\mathbf{H}_t^{\mathcal{T}}} L = \sum_{u \in \mathcal{V}_t} \Bigg(\sum_{v \in \mathcal{N}_t^{\text{walk}}(u)} -log\big(\sigma(< \mathbf{H}_t^{\mathcal{T}}(u), \mathbf{H}_t^{\mathcal{T}}(v) >)\big) - w_{neg} \cdot \sum_{u' \in \mathcal{P}_{neg}^t(u)} log\big(1 - \sigma(< \mathbf{H}_t^{\mathcal{T}}(u'), \mathbf{H}_t^{\mathcal{T}}(u) >)\big) \Bigg) \tag{6}$$

where σ is the sigmoid activation function, $<,>$ is the inner product operation between node representations $\mathbf{H}_t^{\mathcal{T}}(u)$ and $\mathbf{H}_t^{\mathcal{T}}(v)$. $\mathcal{N}_t^{\text{walk}}(u)$ is the set of nodes explored in a fixed length random-walk started at the node u at the t-th graph snapshot $\mathcal{G}_t$. $\mathcal{P}_{neg}^t(u)$ is a negative sampling distribution for the graph snapshot $\mathcal{G}_t$, and w_{neg} is the negative sampling ratio. We optimize the weight parameter matrices in the structural and the temporal attention layers based on the loss function in Equation 6 and the backpropagation algorithm.

B. Distill2Vec-$\mathcal{S}$ - Student Model

To reduce the high online inference latency of the teacher model Distill2Vec-$\mathcal{T}$, we train a compact student model Distill2Vec-$\mathcal{S}$ on the online graph snapshots $\mathcal{G}^{\mathcal{S}}$. For each time step $t = m+1, \ldots, T$, the student model Distill2Vec-$\mathcal{S}$ computes the structural node representations $\mathbf{C}_t(u)$. To capture the graph evolution over the last l consecutive historical graph snapshots $\{\mathcal{G}_{t-l}^{\mathcal{S}}, \ldots, \mathcal{G}_t^{\mathcal{S}}\}$, Distill2Vec-$\mathcal{S}$ computes the temporal node representations $\mathbf{D}_t(u)$. The final node representations $\mathbf{H}_t^{\mathcal{S}}(u)$ are calculated based on Equation 5.

We employ a knowledge distillation strategy on the student model Distill2Vec-$\mathcal{S}$ to transfer the knowledge of the pre-trained teacher model Distill2Vec-$\mathcal{T}$. In practice, the student model Distill2Vec-$\mathcal{S}$ adopts the following distillation loss function $L^{\mathcal{D}}$ during the online training process:

$$\min_{\mathbf{H}^{\mathcal{S}}} L^{\mathcal{D}} = (1-\gamma)L^{\mathcal{S}} + \gamma L^{\mathcal{F}} \tag{7}$$

where $L^{\mathcal{S}}$ is the binary cross-entropy loss that measures the accuracy error of the student model on the online data, and $L^{\mathcal{F}} = \mathcal{KL}(H_t^{\mathcal{S}}(u) \mid H_t^{\mathcal{T}}(u))$ is the Kullback-Leibler (KL) divergence between the node embeddings $\mathbf{H}_t^{\mathcal{S}}(u)$ and $\mathbf{H}_t^{\mathcal{T}}(u)$ for each node $u \in \mathcal{V}_t$ [20]. This means that the student model Distill2Vec-$\mathcal{S}$ mimics the teacher model Distill2Vec-$\mathcal{T}$ during online training, to achieve similar performance with low number of model parameters [7], [10], [11]. Hyperparameter $\gamma \in [0, 1]$ balances the distillation process and the prediction error of the student model Distill2Vec-$\mathcal{S}$ on the online data. High values of γ reflect on generating node embeddings $\mathbf{H}_t^{\mathcal{S}}(u)$ similar to the node embeddings $\mathbf{H}_t^{\mathcal{T}}(u)$ of the student model Distill2Vec-$\mathcal{T}$. Instead, low values of γ emphasize on the prediction errors of the student model Distill2Vec-$\mathcal{S}$. This allows the student model to overcome any bias introduced by the teacher and achieve similar or better performance than Distill2Vec-$\mathcal{T}$ [7], [8], [10], [11].

III. Experiments

A. Evaluation Setup

We evaluate the performance of the proposed distillation strategy based on two publicly available datasets, that is the Yelp[1], with $6,569$ users and businesses and $95,361$ ratings in 16 graph snapshots, and ML-10M[2] with $20,537$ users/movies and $43,760$ user/tag interactions in Movielens and 12 graph snapshots. In our experiments, we train the teacher model Distill2Vec-$\mathcal{T}$ on the offline graph snapshots $\mathcal{G}^{\mathcal{T}}$. For each dataset, we consider the first 5 time steps ($m = 5$) as the offline graph snapshots $\mathcal{G}^{\mathcal{T}}$ and the remaining time steps as the online graph snapshots $\mathcal{G}^{\mathcal{S}}$, that is 11 and 7 test graph snapshots for the Yelp and ML-10M datasets, respectively. We measure the online inference efficiency based on the required number of parameters to train each model. We adopt the Area Under the ROC Curve (AUC), to evaluate the performance of the link prediction task [3], [21]. For each graph snapshot in $\mathcal{G}^{\mathcal{S}}$, we report average AUC values over five randomized runs.

We compare the proposed Distill2Vec-$\mathcal{T}$ and Distill2Vec-$\mathcal{S}$ models with the following baseline strategies: i) DynVGAE[3] [5]; ii) DynamicTriad[4] [2]; iii) TDGNN[5] [22]; iv) DyREP[6] [1]; v) DMTKG-$\mathcal{T}$[7] [23], the teacher model of the knowledge distillation strategy applied on the DeepGraph graph representation learning approach [24]. DMTKG-$\mathcal{T}$ computes the node embeddings on static graphs by employing Convolutional Neural Networks on the intermediate node representations generated by the Heat Kernel Signature (HKS); vi) DMTKG-$\mathcal{S}$ [23], the student model of the DMTKG knowledge distillation

[1] https://www.yelp.com/dataset
[2] https://grouplens.org/datasets/movielens/
[3] https://github.com/stefanosantaris/DynVGAE
[4] https://github.com/luckiezhou/DynamicTriad
[5] https://github.com/stefanosantaris/TDGNN
[6] https://github.com/uoguelph-mlrg/LDG
[7] https://github.com/stefanosantaris/DMTKG

TABLE I
NUMBER OF REQUIRED PARAMETERS IN MILLIONS TO TRAIN EACH MODEL FOR THE ONLINE GRAPH SNAPSHOTS/TIME STEPS

	Yelp							
Time Step	**Distill2Vec-$\mathcal{T}$**	**Distill2Vec-$\mathcal{S}$**	**DynVGAE**	**DynamicTriad**	**TDGNN**	**DyREP**	**DMTKG-$\mathcal{T}$**	**DMTKG-$\mathcal{S}$**
1	1.054	**0.214**	6.090	4.185	2.593	8.295	2.182	1.063
2	1.054	**0.238**	6.649	4.338	2.984	9.235	2.182	1.099
3	1.054	**0.261**	7.187	5.027	3.495	10.591	2.182	1.123
4	1.054	**0.283**	7.685	5.892	3.891	11.058	2.182	1.155
5	1.054	**0.304**	8.225	6.236	4.185	11.837	2.182	1.192
6	1.054	**0.327**	8.809	6.915	4.563	12.293	2.182	1.226
7	1.054	**0.351**	9.380	7.448	4.982	12.944	2.182	1.468
8	1.054	**0.375**	9.933	8.109	5.527	13.284	2.182	1.591
9	1.054	**0.398**	10.308	9.235	6.019	13.749	2.182	1.802
10	1.054	**0.413**	10.658	9.763	6.237	13.987	2.182	1.914
11	1.054	**0.428**	11.236	10.291	6.832	14.235	2.182	2.022
	ML-10M							
1	6.956	**1.542**	6.035	5.923	4.285	10.234	5.293	3.927
2	6.956	**1.700**	6.668	6.142	4.928	11.083	5.293	4.023
3	6.956	**2.011**	7.911	6.591	5.291	12.953	5.293	4.125
4	6.956	**2.127**	8.374	7.839	6.018	13.392	5.293	4.329
5	6.956	**2.264**	8.922	8.113	6.827	14.952	5.293	4.532
6	6.956	**2.375**	9.367	8.788	7.283	15.295	5.293	4.728
7	6.956	**2.562**	10.113	9.423	8.183	16.223	5.293	4.892

TABLE II
AVERAGE AUC FOR EACH ONLINE GRAPH SNAPSHOTS/TIME STEP

	Yelp								
Time Step	**Distill2Vec-$\mathcal{T}$**	**Distill2Vec-$\mathcal{S}$**	**Distill2Vec-$\mathcal{L}$**	**DynVGAE**	**DynamicTriad**	**TDGNN**	**DyREP**	**DMTKG-$\mathcal{T}$**	**DMTKG-$\mathcal{S}$**
1	69.12 ± 0.13	$\mathbf{69.23 \pm 0.12}$	69.13 ± 0.12	62.15 ± 0.21	67.32 ± 0.10	68.14 ± 0.28	64.17 ± 0.05	58.03 ± 0.26	59.42 ± 0.29
2	69.01 ± 0.13	$\mathbf{69.32 \pm 0.11}$	69.15 ± 0.14	62.19 ± 0.23	67.41 ± 0.09	68.23 ± 0.24	64.86 ± 0.04	57.76 ± 0.28	58.72 ± 0.23
3	68.23 ± 0.14	$\mathbf{69.38 \pm 0.11}$	69.19 ± 0.12	62.21 ± 0.19	67.12 ± 0.09	67.53 ± 0.29	65.58 ± 0.02	57.61 ± 0.21	58.94 ± 0.26
4	67.64 ± 0.16	$\mathbf{69.68 \pm 0.14}$	69.21 ± 0.12	62.23 ± 0.25	67.58 ± 0.06	67.64 ± 0.25	65.82 ± 0.06	57.44 ± 0.27	57.83 ± 0.27
5	66.97 ± 0.15	$\mathbf{69.89 \pm 0.11}$	69.25 ± 0.13	62.22 ± 0.25	67.93 ± 0.08	68.18 ± 0.25	65.91 ± 0.02	54.89 ± 0.28	56.28 ± 0.27
6	65.59 ± 0.14	$\mathbf{69.92 \pm 0.11}$	69.27 ± 0.12	62.25 ± 0.24	67.24 ± 0.11	69.19 ± 0.26	66.32 ± 0.05	55.27 ± 0.29	56.63 ± 0.28
7	65.02 ± 0.16	$\mathbf{70.01 \pm 0.10}$	69.32 ± 0.13	62.35 ± 0.23	68.62 ± 0.09	68.76 ± 0.23	66.57 ± 0.04	55.11 ± 0.25	57.69 ± 0.24
8	64.54 ± 0.17	$\mathbf{70.01 \pm 0.11}$	69.36 ± 0.12	62.46 ± 0.24	68.82 ± 0.08	69.09 ± 0.27	66.54 ± 0.06	55.03 ± 0.26	56.14 ± 0.28
9	64.09 ± 0.15	$\mathbf{70.03 \pm 0.11}$	69.41 ± 0.14	62.82 ± 0.25	68.89 ± 0.10	69.06 ± 0.28	67.78 ± 0.06	56.40 ± 0.28	58.68 ± 0.24
10	64.01 ± 0.16	$\mathbf{69.96 \pm 0.12}$	69.52 ± 0.13	62.91 ± 0.21	68.92 ± 0.09	68.58 ± 0.24	67.51 ± 0.07	55.32 ± 0.26	59.49 ± 0.26
11	63.25 ± 0.17	$\mathbf{69.12 \pm 0.13}$	68.84 ± 0.12	63.02 ± 0.22	68.15 ± 0.07	68.26 ± 0.24	66.40 ± 0.09	54.95 ± 0.29	60.19 ± 0.28
	ML-10M								
1	90.94 ± 0.34	$\mathbf{90.95 \pm 0.26}$	90.95 ± 0.12	79.43 ± 0.52	86.63 ± 0.41	88.35 ± 0.52	83.49 ± 0.41	72.42 ± 0.19	73.19 ± 0.17
2	90.42 ± 0.39	$\mathbf{91.53 \pm 0.22}$	90.97 ± 0.13	80.15 ± 0.61	87.86 ± 0.42	88.89 ± 0.51	83.92 ± 0.44	73.64 ± 0.12	75.82 ± 0.16
3	89.92 ± 0.31	$\mathbf{92.68 \pm 0.25}$	91.04 ± 0.13	80.37 ± 0.56	87.91 ± 0.45	89.26 ± 0.52	85.02 ± 0.45	73.82 ± 0.14	75.53 ± 0.18
4	89.84 ± 0.30	$\mathbf{93.26 \pm 0.27}$	91.13 ± 0.11	81.02 ± 0.58	88.23 ± 0.45	90.64 ± 0.55	86.25 ± 0.44	74.03 ± 0.15	75.67 ± 0.14
5	88.69 ± 0.38	$\mathbf{94.14 \pm 0.25}$	92.37 ± 0.11	82.64 ± 0.51	89.56 ± 0.44	92.20 ± 0.52	85.98 ± 0.46	73.76 ± 0.14	75.82 ± 0.15
6	88.29 ± 0.32	$\mathbf{94.47 \pm 0.21}$	92.59 ± 0.12	82.86 ± 0.45	90.86 ± 0.45	92.45 ± 0.53	86.14 ± 0.42	74.21 ± 0.16	75.74 ± 0.16
7	87.58 ± 0.37	$\mathbf{94.69 \pm 0.28}$	92.84 ± 0.12	82.91 ± 0.59	90.94 ± 0.42	92.61 ± 0.52	87.01 ± 0.45	73.97 ± 0.17	76.18 ± 0.17

strategy, that employs a distillation loss function based on the weighted cross entropy; vii) Distill2Vec-$\mathcal{L}$, a variant of the proposed student model, where we replace the Kullback-Leibler divergence $L^{\mathcal{F}}$ in Equation 7 with the binary cross-entropy loss function, as in [25]. In [26], we report the values of the hyper-parameters of each examined model following a cross-validation strategy.

B. Performance Evaluation

In Table I, we report the number of required parameters in millions to train each model over the different online graph snapshots/time steps. As aforementioned in Section II, the teacher models Distill2Vec-$\mathcal{T}$ and DMTKG-$\mathcal{T}$ are trained on the offline data $\mathcal{G}^{\mathcal{T}}$. Therefore, the model sizes of Distill2Vec-$\mathcal{T}$ and DMTKG-$\mathcal{T}$ are not affected during the evaluation of the model on the online data $\mathcal{G}^{\mathcal{S}}$. We observe that Distill2Vec-$\mathcal{S}$ reduces the model size significantly, when compared with the teacher model Distill2Vec-$\mathcal{T}$, achieving averaged compression ratios of 31:100 and 30:100 for the Yelp and ML-10M datasets, respectively. Moreover, Distill2Vec-$\mathcal{S}$ constantly outperforms the baseline approaches in both datasets, in terms of the number of trainable parameters. We omit the number of parameters for Distill2Vec-$\mathcal{L}$, as it is a variant of Distill2Vec-$\mathcal{S}$ with equal number of parameters. The averaged compression ratios of Distill2Vec-$\mathcal{S}$ are 13:100, 16:100, 21:100, 9:100, 27:100 and 37:100, when evaluated against DynVGAE, DynamicTriad, TDGNN, DyREP, DMTKG-$\mathcal{T}$ and DMTKG-$\mathcal{S}$, respectively. The high compression ratios demonstrate the ability of our proposed distillation strategy to significantly reduce the number of model parameters. This means that the proposed student model Distill2Vec-$\mathcal{S}$ achieves low latency during the online inference of the node embeddings, compared with the other baseline approaches. We also notice that DyREP requires a large amount of trainable parameter in both datasets. This indicates that DyREP scales poorly when increasing the

number of nodes in the graph, degrading the performance of the model for online graph snapshots.

In Table II, we evaluate the performance of the student model Distill2Vec-$\mathcal{S}$ against the baseline approaches in the link prediction task. We observe that the student model Distill2Vec-$\mathcal{S}$ constantly outperforms the baseline approaches, in terms of AUC, for both datasets. This indicates that the proposed knowledge distillation strategy can efficiently transfer the knowledge of the pretrained model Distill2Vec-$\mathcal{T}$ to the student model Distill2Vec-$\mathcal{S}$. Therefore, Distill2Vec-$\mathcal{S}$ achieves high link prediction accuracy, while reducing the number of trainable parameters. Moreover, we observe that Distill2Vec-$\mathcal{L}$ exhibits similar behaviour as Distill2Vec-$\mathcal{S}$. However, the cross entropy function in Distill2Vec-$\mathcal{L}$ limits the prediction accuracy, when compared with the Kullback-Leibler divergence of the proposed the Distill2Vec-$\mathcal{S}$ model. Evaluated against TDGNN, which is the second best baseline approach in all datasets, Distill2Vec-$\mathcal{S}$ achieves relative gains 1.8 and 2.5% for the Yelp and ML-10M datasets, respectively. Note that as shown in Table I Distill2Vec-$\mathcal{S}$ achieves average compression ratio of 7:100 and 35:100, in terms of trainable parameters, when compared with TDGNN for the Yelp and ML-10M dataset, respectively. Thus, our model is able to capture the evolution of the graph in the learned node representations, while significantly reducing the model size. In addition, on inspection of Table II we observe that the student models Distill2Vec-$\mathcal{S}$ and DMTKG-$\mathcal{S}$ constantly outperform their respective teacher models Distill2Vec-$\mathcal{T}$ and DMTKG-$\mathcal{T}$. This demonstrates the capability of student models to overcome any bias introduced by the pretrained teacher models on the offline data. Thus, the student model Distill2Vec-$\mathcal{S}$ achieves relative gains of 5.5 and 4.2% against its teacher model for Yelp and ML-10M, respectively.

IV. Conclusion

In this paper, we presented a knowledge distillation strategy to reduce the size of a teacher model for dynamic graph representation learning. The proposed distillation strategy can efficiently generate a compact student model with low online inference latency, while achieving high link prediction accuracy. The experimental results demonstrate the compression efficiency of our distillation strategy. The proposed student model achieves a compression ratio up to 31:100 on two real-world datasets, when compared with the pretrained teacher model. Evaluated against several state-of-the-art approaches, the proposed student model achieves an average relative improvement of 2.2% on both datasets, by significantly reducing the number of required parameters. An interesting future direction is to explore the performance of data-free distillation strategies on dynamic graph representation learning approaches [27]. The main challenge is to design the student model so as to infer accurate embeddings on unobserved nodes by the teacher model.

References

[1] R. Trivedi, M. Farajtabar, P. Biswal, and H. Zha, "Dyrep: Learning representations over dynamic graphs," in *ICLR*, 2019.

[2] L. Zhou, Y. Yang, X. Ren, F. Wu, and Y. Zhuang, "Dynamic network embedding by modeling triadic closure process," in *AAAI*, 2018, pp. 571–578.

[3] A. Sankar, Y. Wu, L. Gou, W. Zhang, and H. Yang, "Dysat: Deep neural representation learning on dynamic graphs via self-attention networks," in *WSDM*, 2020, pp. 519–527.

[4] L. Zhu, D. Guo, J. Yin, G. V. Steeg, and A. Galstyan, "Scalable temporal latent space inference for link prediction in dynamic social networks (extended abstract)," in *ICDE*, 2017, pp. 57–58.

[5] S. Mahdavi, S. Khoshraftar, and A. An, "Dynamic joint variational graph autoencoders," in *ECML*, 2019, pp. 385–401.

[6] E. Hajiramezanali, A. Hasanzadeh, K. R. Narayanan, N. Duffield, M. Zhou, and X. Qian, "Variational graph recurrent neural networks," in *NeurIPS*, 2019, pp. 10 700–10 710.

[7] C. Bucila, R. Caruana, and A. Niculescu-Mizil, "Model compression," in *KDD*, 2006, pp. 535–541.

[8] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *NIPS*, 2015.

[9] Y. Liu, J. Cao, B. Li, C. Yuan, W. Hu, Y. Li, and Y. Duan, "Knowledge distillation via instance relationship graph," in *CVPR*, 2019, pp. 7096–7104.

[10] M. Phuong and C. Lampert, "Towards understanding knowledge distillation," in *ICML*, 2019, pp. 5142–5151.

[11] J. Tang and K. Wang, "Ranking distillation: Learning compact ranking models with high performance for recommender system," in *KDD*, 2018, p. 2289–2298.

[12] H. Li, T. N. Chan, M. L. Yiu, and N. Mamoulis, "Fexipro: Fast and exact inner product retrieval in recommender systems," in *SIGMOD*, 2017, p. 835–850.

[13] Y. Cao, X. Wang, X. He, Z. Hu, and T.-S. Chua, "Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences," in *WWW*, 2019, p. 151–161.

[14] P. Goyal, N. Kamra, X. He, and Y. Liu, "Dyngem: Deep embedding method for dynamic graphs," vol. abs/1805.11273, 2018.

[15] R. Anil, G. Pereyra, A. Passos, R. Ormándi, G. E. Dahl, and G. E. Hinton, "Large scale distributed neural network training through online distillation," in *ICLR*, 2018.

[16] J. Ba and R. Caruana, "Do deep nets really need to be deep?" in *NIPS*, 2014, pp. 2654–2662.

[17] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *ICLR*, 2018.

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[19] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, "Convolutional sequence to sequence learning," in *ICML*, 2017, pp. 1243–1252.

[20] Y. Tian, D. Krishnan, and P. Isola, "Contrastive representation distillation," in *ICLR*, 2020.

[21] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *KDD*, 2016, pp. 855–864.

[22] L. Qu, H. Zhu, Q. Duan, and Y. Shi, "Continuous-time link prediction via temporal dependent graph neural network," in *WWW*, 2020, p. 3026–3032.

[23] J. Ma and Q. Mei, "Graph representation learning via multi-task knowledge distillation," in *NeurIPS*, 2019.

[24] C. Li, X. Guo, and Q. Mei, "Deepgraph: Graph structure predicts network growth," 2016.

[25] S. Antaris, D. Rafailidis, and S. Girdzijauskas, "EGAD: Evolving graph representation learning with self-attention and knowledge distillation for live video streaming events," in *IEEE Big Data*, 2020.

[26] "Supplementary Material," https://github.com/stefanosantaris/Distill2Vec/blob/master/supplementary/supplementary.pdf, 2020, [Online; accessed 24-October-2020].

[27] P. Micaelli and A. J. Storkey, "Zero-shot knowledge transfer via adversarial belief matching," in *NeurIPS*, 2019, pp. 9547–9557.