Unsupervised Inductive Graph-Level Representation Learning via Graph-Graph Proximity

Yunsheng Bai¹, Hao Ding², Yang Qiao¹, Agustin Marinovic¹, Ken Gu¹, Ting Chen¹, Yizhou Sun¹ and Wei Wang¹

¹University of California, Los Angeles
²Purdue University

Abstract

We introduce a novel approach to graph-level representation learning, which is to embed an entire graph into a vector space where the embeddings of two graphs preserve their graph-graph proximity. Our approach, UGRAPH EMB, is a general framework that provides a novel means to performing graph-level embedding in a completely unsupervised and inductive manner. The learned neural network can be considered as a function that receives any graph as input, either seen or unseen in the training set, and transforms it into an embedding. A novel graph-level embedding generation mechanism called Multi-Scale Node Attention (MSNA), is proposed. Experiments on five real graph datasets show that UGRAPH EMB achieves competitive accuracy in the tasks of graph classification, similarity ranking, and graph visualization.

1 Introduction

Recent years we have witnessed the great popularity of graph representation learning with success in not only node-level tasks such as node classification [Kipf and Welling, 2016a] and link prediction [Zhang and Chen, 2018], but also graph-level tasks such as graph classification [Ying et al., 2018] and graph similarity/distance computation [Bai et al., 2019].

There has been a rich body of work [Belkin and Niyogi, 2003; Tang et al., 2015; Qiu et al., 2018] on node-level embeddings that turn each node of a graph into a vector preserving node-node proximity (similarity/distances). Most of these models are unsupervised and demonstrate superb performance in node classification and link prediction. It is natural to raise the question: Can we embed an entire graph into a vector in an unsupervised way, and how? However, most existing methods for graph-level embeddings assume a supervised model [Ying et al., 2018; Zhang and Chen, 2019], with only a few exceptions, such as GRAPH KERNELS [Yanardag and Vishwanathan, 2015] and GRAPH2VEC [Narayanan et al., 2017]. GRAPH KERNELS typically count subgraphs for a given graph and can be slow. GRAPH2VEC is transductive (non-inductive), i.e. it does not naturally generalize to unseen graphs outside the training set.

A key challenge facing designing an unsupervised graph-level embedding model is the lack of graph-level signals in the training stage. Unlike node-level embedding which has a long history in utilizing the link structure of a graph to embed nodes, there lacks such natural proximity (similarity/distance) information between graphs. Supervised methods, therefore, typically resort to graph labels as guidance and use aggregation based methods, e.g. average of node embeddings, to generate graph-level embeddings, with the implicit assumption that good node-level embeddings would automatically lead to good graph-level embeddings using only “intra-graph” information such as node attributes, link structure, etc.

However, this assumption is problematic, as simple aggregation of node embeddings can only preserve limited graph-level properties, which is, however, often insufficient in measuring graph-graph similarity (“inter-graph” information). Inspired by the recent progress on graph proximity modeling [Ktena et al., 2017; Bai et al., 2019], we propose a novel framework, UGRAPH EMB (Unsupervised Graph-level Embedding) that employs multi-scale aggregations of node-level embeddings, guided by the graph-graph proximity defined by well-accepted and domain-agnostic graph proximity metrics such as Graph Edit Distance (GED) [Burke, 1983], Maximum Common Subgraph (MCS) [Burke and Shearer, 1998], etc.

The goal of UGRAPH EMB is to learn high-quality graph-level representations in a completely unsupervised and inductive fashion: During training, it learns a function that maps a graph into a universal embedding space best preserving graph-graph proximity, so that after training, any new graph can be mapped to this embedding space by applying the learned function. Inspired by the recent success of pre-training methods in the text domain, such as ELMo [Peters et al., 2018], BERT [Devlin et al., 2018], and GPT [Radford et al., 2018], we further fine-tune the model via incorporating a supervised loss, to obtain better performance in downstream tasks, including but not limited to:

- **Graph classification.** The embeddings can be fed into any classification model such as logistic regression for graph classification.
- **Graph similarity ranking.** The embeddings learned by UGRAPH EMB preserve the graph-graph proximity by design, and for a graph query, a ranked list of graphs that
are similar to the query can be retrieved.

- **Graph visualization.** The embeddings can be projected into a 2-D space for graph visualization, where each graph is a point. It renders human insights into the dataset and facilitates further database analysis.

In summary, our contributions are three-fold:

1. We formulate the problem of unsupervised inductive graph-level representation learning, and make an initial step towards pre-training methods for graph data. We believe that, given the growing amount of graph datasets of better quality, this work would benefit future works in pre-training methods for graphs.

2. We provide a novel framework, UGRAPHEMB, to generate graph-level embeddings in a completely unsupervised and inductive fashion, well preserving graph proximity. A novel Multi-Scale Node Attention (MSNA) mechanism is proposed to generate graph-level embeddings.

3. We conduct extensive experiments on five real network datasets to demonstrate the superb quality of the embeddings by UGRAPHEMB.

2 The Proposed Approach: UGRAPHEMB

We present the overall architecture of our unsupervised inductive graph-level embedding framework UGRAPHEMB in Figure 1. The key novelty of UGRAPHEMB is the use of graph-graph proximity. To preserve the proximity between two graphs, UGRAPHEMB generates one embedding per graph from node embeddings using a novel mechanism called Multi-Scale Node Attention (MSNA), and computes the proximity using the two graph-level embeddings.

2.1 Inductive Graph-Level Embedding

Node Embedding Generation

For each graph, UGRAPHEMB first generates a set of node embeddings. There are two major properties that the node embedding model has to satisfy:

- **Inductivity.** The model should learn a function such that for any new graph unseen in the training set, the learned function can be applied to the graph, yielding its node embeddings.

- **Permutation-invariance.** The same graph can be represented by different adjacency matrices by permuting the order of nodes, and the node embedding model should not be sensitive to such permutation.

Among various node embedding models, neighbor aggregation methods based on Graph Convolutional Networks (GCN) [Kipf and Welling, 2016a] are permutation-invariant and inductive. The reason is that the core operation, graph convolution, updates the representation of a node by aggregating the embedding of itself and the embeddings of its neighbors. Since the aggregation function treats the neighbors of a node as a set, the order does not affect the result.

A series of neighbor aggregation methods have been proposed with different ways to aggregate neighbor information, e.g. GRAPHSAGE [Hamilton et al., 2017], GAT [Velickovic et al., 2018], GIN [Xu et al., 2019], etc. Since UGRAPHEMB is a general framework for graph-level embeddings, and all these models satisfy the two properties, any one of these methods can be integrated. We therefore adopt the most recent and state-of-the-art method, Graph Isomorphism Network (GIN) [Xu et al., 2019], in our framework:

\[
u_i^{(k)} = \text{MLP}^{(k)}_{W_k} \left( \left( 1 + \epsilon^{(k)} \right) \cdot u_i^{(k-1)} + \sum_{j \in N(i)} u_j^{(k-1)} \right) \tag{1}\]

where \(u_i\) is the representation of node \(i\), \(N(i)\) is the set of neighbors of node \(i\), \(\text{MLP}^{(k)}_{W_k}\) denotes multi-layer perceptrons for the \(k\)-th GIN layer with learnable weights \(W_k\), and \(\epsilon\) is a scalar that can either be learned by gradient descent or be a hyperparameter. GIN has been proven to be theoretically the most powerful GNN under the neighbor aggregation framework [Xu et al., 2019].

Graph Embedding Generation

After node embeddings are generated, UGRAPHEMB generates one embedding per graph using these node embeddings. Existing methods are typically based on aggregating node embeddings, by either a simple sum or average, or some more sophisticated way to aggregate.

However, since our goal is to embed each graph as a single point in the embedding space that preserves graph-graph proximity, the graph embedding generation model should:

- **Capture structural difference at multiple scales.** Applying a neighbor aggregation layer on nodes such as GIN cause the information to flow from a node to its direct neighbors, so sequentially stacking \(K\) layers would cause the final representation of a node to include information from its \(K\)-th order neighbors. However, after many neighbor aggregation layers, the learned embeddings could be too coarse to capture the structural difference in small local regions between two similar graphs. Capturing structural difference at multiple scales is therefore important for UGRAPHEMB to generate high-quality graph-level embeddings.

- **Be adaptive to different graph proximity metrics.** UGRAPHEMB is a general framework that should be able to preserve the graph-graph proximity under any graph proximity metric, such as GED and MCS. A simple aggregation of node embeddings without any learnable parameters limits the expressive power of existing graph-level embedding models.

To tackle both challenges in the graph embedding generation layer, we propose the following Multi-Scale Node Attention (MSNA) mechanism. Denote the input node embeddings of graph \(G\) as \(U_g \in \mathbb{R}^{N \times D}\), where the \(n\)-th row, \(u_n \in \mathbb{R}^D\) is the embedding of node \(n\). The graph level embedding is obtained as follows:

\[
h_g = \text{MLP}_W \left( \prod_{k=1}^{K} \text{ATT}_{\Theta^{(k)}}(U_g) \right) \tag{2}\]
where $\| \|$ denotes concatenation, $K$ denotes the number of neighbor aggregation layers, ATT denotes the following multi-head attention mechanism that transforms node embeddings into a graph-level embedding, and MLP$_W$ denotes multi-layer perceptrons with learnable weights $W$ applied on the concatenated attention results.

The intuition behind Equation 2 is that, instead of only using the node embeddings generated by the last neighbor aggregation layer, we use the node embeddings generated by each of the $K$ neighbor aggregation layers.

ATT is defined as follows:

$$
\text{ATT}_\theta(U_{G}) = \sum_{n=1}^{N} \sigma(u_n^T \text{ReLU}(\Theta(\frac{1}{N} \sum_{m=1}^{N} u_m))) u_n. \tag{3}
$$

where $N$ is the number of nodes, $\sigma$ is the sigmoid function $\sigma(x) = \frac{1}{1+\exp(-x)}$, and $\Theta(k) \in \mathbb{R}^{D \times D}$ is the weight parameters for the $k$-th node embedding layer.

The intuition behind Equation 3 is that, during the generation of graph-level embeddings, the attention weight assigned to each node should be adaptive to the graph proximity metric. To achieve that, the weight is determined by both the node embedding $u_n$, and a learnable graph representation. The learnable graph representation is adaptive to a particular graph proximity via the learnable weight matrix $\Theta(k)$.

### 2.2 Unsupervised Loss via Inter-Graph Proximity Preservation

**Definition of Graph Proximity**

The key novelty of UGRAPHEMB is the use of graph-graph proximity. It is important to select an appropriate graph proximity (similarity/distance) metric. We identify three categories of candidates:

- **Proximity defined by graph labels.**
  
  For graphs that come with labels, we may treat graphs of the same label to be similar to each other. However, such proximity metric may be too coarse, unable to distinguish between graphs of the same label.

- **Proximity given by domain knowledge or human experts.**

  For example, in drug-drug interaction detection [Ma et al., 2018], a domain-specific metric to encode compound chemical structure can be used to compute the similarities between chemical graphs. However, such metrics do not generalize to graphs in other domains. Sometimes, this information may be very expensive to obtain. For example, to measure brain network similarities, a domain-specific preprocessing pipeline involving skull stripping, band-pass filtering, etc. is needed. The final dataset only contains networks from 871 humans [Kien et al., 2017].

- **Proximity defined by domain-agnostic and well-accepted metrics.**

  Metrics such as Graph Edit Distance (GED) [Bunke, 1983] and Maximum Common Subgraph (MCS) [Bunke and Shearer, 1998] have been widely adopted in graph database search [Yan et al., 2005; Liang and Zhao, 2017], are well-defined and general to any domain.

In this paper, we use GED as an example metric to demonstrate UGRAPHEMB. GED measures the minimum number of edit operations to transform one graph to the other, where an edit operation on a graph is an insertion or deletion of a node/edge or relabelling of a node. Thus, the GED metric takes both the graph structure and the node labels/attributes into account. The supplementary material contain more details on GED.

**Prediction of Graph Proximity**

Once the proximity metric is defined, and the graph-level embeddings for $G_i$ and $G_j$ are obtained, denoted as $h_{G_i}$ and $h_{G_j}$, we can compute the similarity/distance between the two graphs.
Multidimensional scaling (MDS) is a classic form of dimensionality reduction [Williams, 2001]. The idea is to embed data points in a low dimensional space so that their pairwise distances are preserved, e.g. via minimizing the loss function

\[ L(h_i, h_j, d_{ij}) = (\| h_i - h_j \|^2 - d_{ij}^2)^2 \]  

(4)

where \( h_i \) and \( h_j \) are the embeddings of points \( i \) and \( j \), and \( d_{ij} \) is their distance.

Since GED is a well-defined graph distance metric, we can minimize the difference between the predicted distance and the ground-truth distance:

\[ L = \mathbb{E}_{(i,j) \sim D}(d_{ij} - \hat{d}_{ij})^2 \]  

(5)

\[ = \mathbb{E}_{(i,j) \sim D}(\| h_{G_i} - h_{G_j} \|^2 - d_{ij})^2. \]  

(6)

where \((i,j)\) is a pair graph sampled from the training set and \(d_{ij}\) is the GED between them.

Alternatively, if the metric is similarity, such as in the case of MCS, we can use the following loss function:

\[ L = \mathbb{E}_{(i,j) \sim D}(s_{ij} - \hat{s}_{ij})^2 \]  

(7)

\[ = \mathbb{E}_{(i,j) \sim D}(h_{G_i}^T h_{G_j} - s_{ij})^2. \]  

(8)

After training, the learned neural network can be applied to any graph, and the graph-level embeddings can facilitate a series of downstream tasks, and can be fine-tuned for specific tasks. For example, for graph classification, a supervised loss function can be used to further enhance the performance.

### 3 Experiments

We evaluate our model, UGRAPHEMB, against a number of state-of-the-art approaches designed for unsupervised node and graph embeddings, to answer the following questions:

**Q1** How superb are the graph-level embeddings generated by UGRAPHEMB, when evaluated with downstream tasks including graph classification and similarity ranking?

**Q2** Do the graph-level embeddings generated by UGRAPHEMB provide meaningful visualization for the graphs in a graph database?

**Q3** Is the quality of the embeddings generated by UGRAPHEMB sensitive to choices of hyperparameters?

#### Datasets

We evaluate the methods on five real graph datasets, PTC, IMDBM, WEB, NC1109, and REDDIT12K. The largest dataset, REDDIT12K, has 11929 graphs.

#### 3.1 Task 1: Graph Classification

Intuitively, the higher the quality of the embeddings, the better the classification accuracy. Thus, we feed the graph-level embeddings generated by UGRAPHEMB and the baselines into a logistic regression classifier to evaluate the quality: (1) GRAPH KERNELES (GRAPHLET (GK), DEEP GRAPHLET (DGK), SHORTEST PATH (SP), DEEP SHORTEST PATH (DSP), WEISFEILER-LEHMAN (WL), and DEEP WEISFEILER-LEHMAN (DWL)); (2) GRAPH2VEC [Narayanan et al., 2017]; (3) NETMF [Qiu et al., 2018]; (4) GRAPH-SAGE [Hamilton et al., 2017].

For GRAPH KERNELES, we also try using the kernel matrix and SVM classifier as it is the standard procedure outlined in [Yanardag and Vishwanathan, 2015], and report the better accuracy of the two. For (3) and (4), we try different averaging schemes on node embeddings to obtain the graph-level embeddings and report their best accuracy.

As shown in Table 1, UGRAPHEMB without fine-tuning, i.e. using only the unsupervised “inter-graph” information, can already achieve top 2 on 3 out of 5 datasets and demonstrates competitive accuracy on the other datasets. With fine-tuning (UGRAPHEMB-F), our model can achieve the best result on 4 out of 5 datasets. Methods specifically designed for graph-level embeddings (GRAPH KERNELES, GRAPH2VEC, and UGRAPHEMB) consistently outperform methods designed for node-level embeddings (NETMF and GRAPH-SAGE), suggesting that good node-level embeddings do not naturally imply good graph-level representations.

#### 3.2 Task 2: Similarity Ranking

For each dataset, we split it into training, validation, and testing sets by 6:2:2, and report the averaged Mean Squared Error (mse), Kendall’s Rank Correlation Coefficient (\(\tau\)) [Kendall, 1938], and Precision at 10 (p@10) to test the ranking performance.

Table 2 shows that UGRAPHEMB achieves state-of-the-art ranking performance under all settings except one. This should not be a surprise, because only UGRAPHEMB utilizes the ground-truth GED results collectively determined by BEAM [Neuhaus et al., 2006], HUNGARIAN [Riesen and Bunke, 2009], and VJ [Fankhauser et al., 2011]. UGRAPHEMB even outperforms HED [Fischer et al., 2015], a state-of-the-art approximate GED computation algorithm, under most settings, further confirming its strong ability to generate proximity-preserving graph embeddings by learning from a specific graph proximity metric, which is GED in this case.

#### 3.3 Task 3: Embedding Visualization

Visualizing the embeddings on a two-dimensional space is a popular way to evaluate node embedding methods [Tang et al., 2015]. However, we are among the first to investigate the question: Are the graph-level embeddings generated by a model like UGRAPHEMB provide meaningful visualization?

We feed the graph embeddings learned by all the methods into the visualization tool t-SNE [Maaten and Hinton, 2008]. The three deep graph kernels, i.e. DGK, DSP, and WDL,
generate the same embeddings as the non-deep versions, but use additional techniques [Yanardag and Vishwanathan, 2015] to modify the similarity kernel matrices, resulting in different classification and ranking performance.

From Figure 2, we can see that UGRAPHEMB can separate the graphs in IMDBM into multiple clusters, where graphs in each cluster share some common substructures.

Such clustering effect is likely due to our use of graph-graph proximity scores, and is thus not observed in NetMF or GraphSAGE. For Graph Kernels and Graph2Vec though, there are indeed clustering effects, but by examining the actual graphs, we can see that graph-graph proximity is not well-preserved by their clusters (e.g. for WL graph 1, 2 and 9 should be close to each other; for Graph2Vec, graph 1, 2, and 12 should be close to each other), explaining their worse similarity ranking performance in Table 2 compared to UGRAPHEMB.

### 3.4 Parameter Sensitivity of UGRAPHEMB

We evaluate how the dimension of the graph-level embeddings and the percentage of graph pairs with ground-truth GEDs used to train the model can affect the results. We report the graph classification accuracy on IMDBM.

As can be seen in Figure 3, the performance becomes marginally better if larger dimensions are used. For the percentage of training pairs with ground-truth GEDs, the performance drops as less pairs are used. Note that the x-axis is in log-scale. When we only use 0.001% of all the training graph pairs (only 8 pairs with ground-truth GEDs), the performance is still better than many baseline methods, exhibiting impressive insensitivity to data sparsity.

### 4 Related Work

Unsupervised graph representation learning has a long history. Classic works including NetMF [Qu et al., 2018], LINE [Tang et al., 2015], DeepWalk [Perozzi et al., 2014], etc., which typically generate an embedding for each node in one graph. Theoretical analysis shows that many of these works cannot handle embeddings for multiple graphs in the sense that the node embeddings in one graph are not comparable to those in another graph in any straightforward way [Heimann and Koutra, 2017]. A simple permutation of node indices could cause the node embedding to be very different.

More recently, some of the methods based on Graph Convolutional Networks (GCN) [Defferrard et al., 2016; Kipf and Welling, 2016a], such as VGAE [Kipf and Welling, 2016b], satisfy the desired permutation-invariance property. Categorized as "graph autoencoders" [Wu et al., 2019], they also belong to the family of graph neural network methods. Although satisfying the permutation-invariance requirement, these autoencoders are still designed to generate unsupervised node embeddings.

Methods designed for unsupervised graph-level embeddings include Graph2Vec [Narayanan et al., 2017] and Graph Kernels [Yanardag and Vishwanathan, 2015], which however are either not based on learning or not inductive. Unlike node-level information which is reflected in the neighborhood of a node, graph-level information is much more limited. A large amount of graph neural network models resort to graph labels as a source of such information, making the models supervised aiming to improve graph classification accuracy specifically, such as DiffPool [Ying et al., 2018], Caps-GNN [Zhang and Chen, 2019], etc., while UGRAPHEMB learns a function that maps each graph into an embedding that can be used to facilitate many downstream tasks.

### 5 Conclusion

We present UGRAPHEMB, an end-to-end neural network based framework aiming to embed an entire graph into an embedding preserving the proximity between graphs in the dataset under a graph proximity metric, such as Graph Edit Distance (GED). A novel mechanism for generating graph-
level embeddings is proposed. Experiments show that the produced graph-level embeddings achieve competitive performance on three downstream tasks: graph classification, similarity ranking, and graph visualization.

Acknowledgements

This work is partially supported by NIH R01GM115833 and U01HG008488, NSF DBI-1565137, DGE-1829071, NSF III-1705169, NSF CAREER Award 1741634, and Amazon Research Award.

References


[Devlin et al., 2018] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of
deep bidirectional transformers for language understanding. 


[Qiu et al., 2018] Jiezhihong Qiu, Yuxiao Dong, Hao Ma, Jian Li, KuanSan Wang, and Jie Tang. Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec. WSDM, 2018.


[Yan et al., 2005] Xifeng Yan, Philip S Yu, and Jiawei Han. Substructure similarity search in graph databases. In SIGMOD, pages 766–777. ACM, 2005.


