Semi-Supervised Deep Learning for Multiplex Networks

Anasua Mitra
anasua.mitra@iitg.ac.in
Indian Institute of Technology
Guwahati, India

Priyesh Vijayan
priyesh.vijayan@mail.mcgill.ca
McGill University & Mila
Canada

Ranbir Sanasam
ranbir@iitg.ac.in
Indian Institute of Technology
Guwahati, India

Diganta Goswami
dgoswami@iitg.ac.in
Indian Institute of Technology
Guwahati, India

Srinivasan Parthasarathy
srini@cse.ohio-state.edu
Ohio State University
USA

Balaraman Ravindran
ravi@cse.iitm.ac.in
Robert Bosch Centre for Data Science
and AI, Indian Institute of Technology
Madras, India

ABSTRACT

Multiplex networks are complex graph structures in which a set of entities are connected to each other via multiple types of relations, each relation representing a distinct layer. Such graphs are used to investigate many complex biological, social, and technological systems. In this work, we present a novel semi-supervised approach for structure-aware representation learning on multiplex networks. Our approach relies on maximizing the mutual information between local node-wise patch representations and label correlated structure-aware global graph representations to model the nodes and cluster structures jointly. Specifically, it leverages a novel cluster-aware, node-contextualized global graph summary generation strategy for effective joint-modeling of node and cluster representations across the layers of a multiplex network. Empirically, we demonstrate that the proposed architecture outperforms state-of-the-art methods in a range of tasks: classification, clustering, visualization, and similarity search on seven real-world multiplex networks for various experiment settings.

CCS CONCEPTS

• **Computing methodologies** → **Semi-supervised learning settings**; **Neural networks**; **Learning latent representations**.

KEYWORDS

Multiplex networks; infomax principle; network embedding

ACM Reference Format:
Anasua Mitra, Priyesh Vijayan, Ranbir Sanasam, Diganta Goswami, Srinivasan Parthasarathy, and Balaraman Ravindran. 2021. Semi-Supervised Deep Learning for Multiplex Networks. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '21), August 14–18, 2021, Virtual Event, Singapore.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3447548.3467443

KDD '21, August 14–18, 2021, Virtual Event, Singapore

ACM ISBN 978-1-4503-8332-5/21/08…$15.00
https://doi.org/10.1145/3447548.3467443

1 INTRODUCTION

Entities in many real-world problems are related to each other in multiple ways. Such relations are often modeled as graph-structured data where the nodes represent entities, and edges between a pair of nodes represent the interactions between the entities. Learning representations for such networked data to mine, analyze and build predictive models has been gaining a lot of traction recently with the advent of deep learning-based network embedding models [6, 32].

Increasingly such relations are complex, with multiple relationship types linking entities. Such networked data are often naturally represented as multi-layered graphs [9], where each component layer focuses on a specific relation type and can involve different sets of nodes. In this work, we focus on *Multiplex networks*, a special case of multi-layer networks where the graphs in all the layers share the same set of nodes with distinct relations in different layers. Such multiplex network structures are observed in numerous environments such as bibliographic networks, temporal networks, traffic networks, brain networks, protein-drug-disease interaction, etc. The involvement of the same set of nodes across multiple types of relations, distinctive structures in different layers, and the interplay among various layers of networks — makes representation learning of multiplex networks a challenging task.

Existing multiplex Network Representation Learning (NRL) methods learn node embeddings that encode the local relational structure of nodes by using graph convolutions [4, 15, 20] or random walks [14, 33] within a subgraph centered at the node of interest. Though there are many powerful models to learn local structures, only a few works encode global structures [15, 20] even in the case of the more widely researched simple homogeneous graphs. Global structural information is encoded in representation learning models through one of three approaches: (i) clustering constraints [26, 30]; (ii) auto-encoding objectives on the adjacency matrix [26, 29, 30] or node embeddings [3]; and (iii) Mutual Information Maximization (InfoMax) [25, 28] objectives that maximize the Mutual Information (MI) between the representations of local nodes and the global summary of the graph derived from the local contexts of all the nodes [20, 23, 25, 28].

Clustering constraints are also often realized with auto-encoding objectives, that in general, are challenging to scale [26, 30]. In contrast to the first two methods, the InfoMax based approaches use Graph Neural Networks (GNNs) to obtain both local and global context and are potentially more scalable [20, 23, 28]. However, InfoMax objectives that encode global information assume a shared

global graph context for all the nodes despite the fact that, in most cases, every node has a different global structure rooted at each node. This calls for a different contextualized global graph representation for each node (analogous to the notion of personalization).

Contributions. In this work, we propose the first node-contextualized InfoMax based semi-supervised learning architecture for multiplex networks. The primary contributions of our work are:

- Motivated by the need for contextualized global graph representations, we propose a novel *joint* node and cluster representation learning model that defines a structure-aware intra-layer graph context for a node.
- In the semi-supervised setting, the cluster constraints are provided by the partial label information and are shared across layers to learn similar clusters across relational layers in terms of label correlations. To facilitate this, we further constrain the nodes connected by cross-edges to have similar embeddings, thereby indirectly influencing the layerwise InfoMax objective to capture global cluster information across multiple layers.
- We evaluate the model on seven multiplex networks for node classification, clustering, and similarity search. Our proposed model achieves the best overall performance outperforming state-of-the-art methods like DMGI [20], MGCN [4], HAN [31].
- Also, the learned node embeddings lead to well-separated homogeneous clusters in t-SNE visualizations.

2 BACKGROUND AND KEY INTUITION

2.1 Notation and Problem Statement

Notation: Let the multi-layer representation of a multiplex network with vertex set, $\mathcal{V}$ and relation set, $\mathcal{R}$ such that $|\mathcal{R}| > 1$ be defined by an $|\mathcal{R}|$-layered graph, $\mathcal{G} = \{\mathcal{G}_1, \mathcal{G}_2, ..., \mathcal{G}_{|\mathcal{R}|}\}$, where $\mathcal{G}_r = (\mathcal{V}, A_r)$ with A_r being the adjacency matrix for the r^{th} layer corresponding to the r^{th} relation. We generalize the adjacency matrix notation to include both intra-layer and inter-layer edges between nodes in different layers by letting $A_{(r,r)}$ to denote r^{th} layer's adjacency matrix corresponding to its intra-layer edges and $A_{(r,s)}$ to denote the inter-layer edges between layer r and s when $r \neq s$. Note that $A_{(r,r)}, A_{(r,s)} \in \mathbb{R}_+^{|\mathcal{V}|\times|\mathcal{V}|}$ as all the layers share the same set of nodes, $\mathcal{V}$. Often, the nodes are associated with a feature set, $\mathcal{F}$ and the node feature matrix is denoted as $X \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{F}|}$.

Semi-Supervised Learning Task: Given a multiplex graph, $\mathcal{G} = (\mathcal{V}, A, X)$, a label set, $\mathcal{Q}$ and set of labeled nodes, $\mathcal{L}$ with ground truth label assignment matrix, $Y \in \{0,1\}^{|\mathcal{V}|\times|\mathcal{Q}|}$, the task is to predict labels for all unlabeled nodes, $\mathcal{U} = \mathcal{V} \setminus \mathcal{L}$. For efficient Semi-Supervised Learning (SSL) on multiplex networks, it is essential to learn a low d-dimensional ($d \ll |\mathcal{F}|$) node embedding, $Z \in \mathbb{R}^{|\mathcal{V}|\times d}$ that encodes relevant structural and label correlation information within and across layers, useful for downstream machine learning tasks such as node classification and clustering.

2.2 Multiplex NRL and InfoMax Objective

Node Representation Learning: Multiplex Network Representation Learning methods encode useful information for all the nodes into a low d-dimensional node embedding, $Z_r \in \mathbb{R}^{|\mathcal{V}|\times d}$ for each layer r and then aggregate information across layer by leveraging the cross edges, into a joint embedding, $Z \in \mathbb{R}^{|\mathcal{V}|\times d}$.

Graph Convolutional Networks (GCNs) [8, 24] are widely used node embedding architectures that encode attribute-based local structural information from a node's multi-hop neighborhood. In the context of multiplex networks, GCNs [20, 28, 36] are used layer-wise to obtain node embeddings based on the intra-layer edges. Then, to get a joint node embedding, embeddings from different node counterparts across layers are aggregated via the cross-edges.

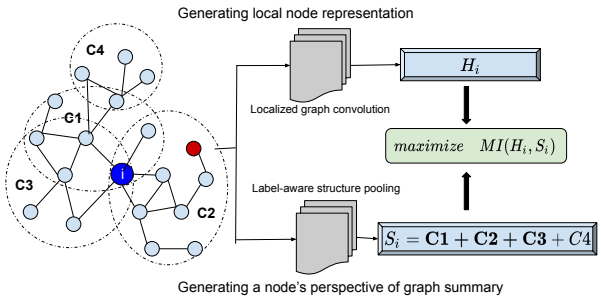


Figure 1: Label correlated structure-aware InfoMax

Encoding Global Information with InfoMax based objective: While GCNs are powerful models to encode local structures, they do not encode global contextual information. On that front, recent efforts in the NRL community have adopted the Mutual Information Maximization (InfoMax) objective, initially proposed in image feature extraction pipelines for learning structurally dependent rich local and global representations of images — to the graph domain to learn rich node [20, 28] and graph-level representations [25]. In problems where the data is sampled from a set of graphs, each data instance is a graph, and the task is to learn a global graph representation wherein the InfoMax based models learn graph representations by maximizing their mutual information (MI) with the local node representations [25]. In this work, we are interested in the semi-supervised transductive setting. Given a partially labeled graph, we look to learn node representations that allow us to predict labels for the rest of the nodes in the same graph. In this setting, existing Infomax models learn the local node representations by maximizing its MI with a single global graph representation [20, 28]. As we argue in the next section, where we describe this work's intuition, this single global representation is often inadequate.

From the computational aspect, the mutual information between two variables can be maximized by leveraging the KL-divergence between their Joint distribution and the product of marginals. However, since estimating MI in high dimension and continuous data is complex, in practice, scalable Neural MI estimators [1] that maximize a tractable lower-bound are used. *Noise Contrastive Estimation-based (NCE) loss* that discriminates samples from a true joint distribution against a noisy product of marginals (negative samples) are simple yet effective ways to realize this lower bound. It can be viewed from the predictive coding perspective, where given a global-whole representation, the task is to predict a corresponding local-part representation. This forces the discriminator to provide a high score to a related pair of local-global representations compared to unrelated pairs. Henceforth, unless specified otherwise, we adopt minimizing the NCE loss for this purpose.

2.3 Key Intuition

In a typical InfoMax based NRL setup, the global context is defined by all nodes in the graph. Thus, each node in the graph does not have its own contextual view of the graph. Instead has a shared global context that is the same for all the nodes, *even though they may be structurally connected differently within the graph.* For example, from the graph in Fig: 1, the red node and blue node belonging to the same cluster C2 will have different non-local network measures such as betweenness measure, participation coefficient [5], etc., as the structure of the (sub)graph centered around these nodes are differently connected to the rest of the graph since one is in the center of a cluster, and other is at the end of a whisker.

When a naive global graph summary function such as the average of all node embeddings is used, the global context for all nodes becomes the same as their global context is isomorphic. Naively maximizing the MI of a nodes' local representations with a shared global context might bias the model to encode trivial and noisy information found across all the nodes' local information. Albeit, naively defining a different global context for each node, such as a sub-graph-based approach, will shoot down the original objective of learning useful shared information from across the graph.

Thus, this calls for a careful design of a *contextualized representation of shared global information* that facilitates encoding relevant non-trivial shared information present across the graph when maximizing the MI with the local node information. In Figure 1, even though the example graph has many clusters, only $C1, C2, C3$ are relevant to the blue node i. Therefore, the global context for node i should be more inclined towards $C1, C2, C3$ instead of a naive summary of all candidate clusters. In light of this simple intuition and motivated by participation scores, we propose a cluster-based InfoMax objective to learn node representations. *The clusters encode shared global graph information, and the node-specific global context is obtained by aggregating information from the clusters with which the node is associated.* In particular, for the semi-supervised classification task, we define label-correlated structure-aware clusters that jointly learn node and cluster representations by optimizing the InfoMax principle.

3 PROPOSED METHODOLOGY

In this section, we explain step-by-step our proposed approach to learning node representations for multiplex networks. The proposed method Semi-Supervised Deep Clustered Multiplex (SSDCM) in Figure 2 — (i) learns relation-specific node representation that encodes both local and global information, (ii) enforces cross-edge based regularization to align all nodes connected across layers to lie on the same space, then (iii) learns a joint embedding across layers for all nodes through a consensus regularization and (iv) finally enables label predictions with this joint embedding.

3.1 Learning Node Representations

The first component of our model learns relation-specific ($\mathcal{R}$) local node representations, U_r. Then these learned node representations are made aware of their individual global context, S_r which summarizes graph level information. We do this by maximizing the mutual information between them and this can be realized by minimizing a noise-contrastive estimation such as a binary cross-entropy loss as provided in the equation below,

$$\mathfrak{O}_{MI} = \sum_{r\in\mathcal{R}}\sum_{i\in\mathcal{V}}\Big(\log(\mathfrak{D}(U_r^i, S_r^i)) + \sum_{j=1}^{N}\log(1-\mathfrak{D}(\tilde{U}_r^j, S_r^i))\Big) \quad (1)$$

where $\mathfrak{D}: \mathbb{R}^{2d} \mapsto \mathbb{R}$ is a discriminator function that assigns a probability score to a pair of local-global node representations using a bi-linear scoring matrix $B \in \mathbb{R}^{d\times d}$, i.e., $\mathfrak{D}(U_r^i, S_r^i) = \sigma(U_r^{iT} B S_r^i)$, σ being the sigmoid non-linearity. Similar to [20], we learn this discriminator universally, i.e., the weight is shared across all layers with an intention to capture local-global representation correlations across the relations. The discriminator gives a local-global summary a higher value if the local summary is highly relevant to the global summary and, in contrast, assigns a lower score if the local summary is less relevant or irrelevant to the global summary. For every node i in each relation $r \in \mathcal{R}$, N negative local summaries are paired with that node's contextual global summary to train the discriminator $\mathfrak{D}$. Following [28], we create corrupted local patches $\tilde{U}_r^j$ for each relation r by row-shuffling the node features X and passing it through the same local structure encoder.

Having explained the overall structure of our InfoMax objective, we get into the details of how to learn (a) local node representations, (b) global node representations, and (c) the clustering strategy that provides the global context for nodes.

3.1.1 Local Node Representations. For each relation $r \in \mathcal{R}$, we obtain an M-hop local node representations $U_r \in \mathbb{R}^{|\mathcal{V}|\times d}$ with a Graph Convolutional Neural Network (GCN) encoder $\mathfrak{E}_\mathbf{r}$. GCNs obtain an M-hop local representation by recursively propagating aggregated neighborhood information. Let $\tilde{A}_{(r,r)} = A_{(r,r)} + \epsilon I_{|\mathcal{V}|}$ be the intra-layer adjacency matrix for relation r with added ϵ-weighted self-loops (similar to a Random Walk with Restart (RWR) probability kernel). Here, we use the normalized adjacency matrix, $\hat{A}_{(r,r)} = (\tilde{D}_{(r,r)}^{-\frac{1}{2}}\tilde{A}_{(r,r)}\tilde{D}_{(r,r)}^{-\frac{1}{2}})$, as the GCN's neighborhood aggregation kernel, where $\tilde{D}_{(r,r)}$ is the diagonal degree matrix of $\tilde{A}_{(r,r)}$. An M-hop node embedding is obtained by stacking M-layers of GCNs as in Eqn: 2. The input to the m^{th} GCN layer is the output of the $(m-1)^{\text{th}}$ GCN layer, X_r^{m-1}, with the original node features X fed in as input to the first layer.

$$\begin{aligned} X_r^0 &= X \\ X_r^m &= PReLU(\hat{A}_{(r,r)} X_r^{m-1} W_r^m) \\ U_r &= X_r^M \end{aligned} \quad (2)$$

where W_r^m is learnable weight matrix for the m^{th} GCN layer corresponding to the r^{th} multiplex layer. Assuming all GCN layers' outputs to be of the same dimension d, we have $X_r^m \in \mathbb{R}^{|\mathcal{V}|\times d}$ and $W_r^m \in \mathbb{R}^{d\times d}$, except for the first GCN layer, whose weights are $W_r^0 \in \mathbb{R}^{|\mathcal{F}|\times d}$. The final M-hop GCN representation for each relation, r is treated as that relation's local node embedding, $U_r = X_r^M$.

3.1.2 Contextual Global Node Representations. We learn a contextualized global summary representation, S_r^i for each node i, and for every relation $r \in \mathcal{R}$. In this work, we first capture a global graph level summary by learning K clusters in each relation. Then we leverage these learned clusters to provide a contextual global node summary for all the nodes based on the learned node-cluster

associations. We explain the steps in a top-down manner. We first explain how we obtain a contextualized global graph representation given clustering information, and in the following subsection, we explain how to obtain the clusters.

Across all multiplex layers, we learn K clusters in each relation $r \in \mathcal{R}$. We encode the clustering information with relation-specific K cluster embeddings, $C_r = \{C_r^1, C_r^2, ..., C_r^K\}$ with $C_r^k \in \mathbb{R}^{1\times d}$ and node-cluster assignment matrix, $H_r \in \mathbb{R}^{|\mathcal{V}|\times K}$. Given the learned relation-wise clustering information (C_r, H_r) and local node representations, U_r, we compute the contextual global node representation for a node i as a linear combination of different cluster embeddings, C_r^k weighted by that node's cluster association scores $H_r^i[k], \forall k \in [1, K]$ as mentioned in below.

$$S_r^i = \sum_{k=1}^{K} H_r^i[k] C_r^k \tag{3}$$

3.1.3 Clustering. We now describe how to learn clusters that capture useful global information for a node across all relations. Specifically, we aim to capture globally relevant label information that can enrich local node representations for the semi-supervised node classification task when jointly optimized for the MI between them across relations. To achieve this, we adapt Mitra et al. [18]'s Non-Negative Matrix Factorization formulation to learn label-correlated clusters to a Neural Network setup as follows.

Cluster Embedding: We randomly initialize the set of cluster embeddings C_r for each relation r and allow them to be updated based on the gradients from the model's loss.

Cluster Assignment: We obtain the node-cluster assignment matrix, H_r, by computing the inner-product between node embeddings and cluster embeddings. We then pass it through a softmax layer to obtain normalized probability scores of cluster-memberships for each node, see Eqn: 4.

$$H_r^i[k] = SoftMax(U_r^i . {C_r^k}^T) \tag{4}$$

Non-overlapping Clustering Constraint: To enforce hard cluster membership assignments, we regularize the cluster assignment H_r with block-diagonal constraints. Specifically, we ensure the block size to be one, and the resulting orthogonality constraint enforces less overlap in node assignments between every pair of clusters. This constraint is expressed as a loss function below.

$$\mathfrak{O}_{Orthogonal} = \|H_r^T H_r - I_K\|_F^2 \tag{5}$$

Global Label-homogeneous Clustering Constraint: To capture globally relevant information for each relational graph, we group nodes based on an aspect — enforcing homogeneity within clusters. Precisely, we capture global label-correlation information with a similarity kernel, $\mathcal{S} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$ and cluster nodes according to it. The label similarity kernel is defined between the labeled nodes $\mathcal{L}$ as $\mathcal{S} = Y_{[\mathcal{L}]}.{Y_{[\mathcal{L}]}}^T$. We use a masking strategy to consider only the label information of training nodes for enforcing this.

We now use a Laplacian regularizer to enforce smoothness on the cluster-assignments according to the label-similarity kernel $\mathcal{S}$ as given in the equation below,

$$\mathfrak{O}_{Learn} = Tr(H_r^T \Delta(\mathcal{S}) H_r) \tag{6}$$

where $\Delta(\mathcal{S})$ is the un-normalized Laplacian of the similarity kernel. The above Laplacian smoothing constraint enforces nodes with similar labels to lie in the same/similar clusters.

Note that since this is shared across relations, it enforces similar clustering to be learned across relations. The learned clusters can still vary based on the relation-specific node embeddings, thus capturing global shared context across diverse graph structures. More importantly, notice that the label-similarity kernel can connect nodes that may be far away by a distance longer than the local (M) multi-hop context considered and even can connect two nodes that are not reachable from each other.

In the entire pipeline, cluster learning is facilitated by the following loss function: $\mathfrak{O}_{Clus} = (\mathfrak{O}_{Learn} + \mathfrak{O}_{Orthogonal})$

3.2 Cross-relation Regularization

Since each multiplex layer encodes a different relational aspect of nodes, it is not straightforward to treat the inter-layer edges the same way as intra-layer edges to aggregate information from cross-linked neighbors. Also, the representation for nodes in different layers lies in different spaces and is not compatible with a naive aggregation of information via cross-edges.

Previous works, incorporated inter-layer (cross-graph) edge information into the learning procedure by adopting either cross-graph node embedding regularization [12, 19] or clustering techniques [15]. Since, in our case, we have the same clustering constraints enforced across layers, we opt to regularize embeddings of nodes connected by cross edges to lie in the same space.

$$\mathfrak{O}_{Cross} = \sum_{r,s\in\mathcal{R}} ||O_{(r,s)} U_r - A_{(r,s)} U_s||_F^2 \tag{7}$$

where $O_{(r,s)} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$ is a binary diagonal matrix, with $O_{(r,s)}^{i,i} = 0$ if the corresponding $A_{(r,s)}^i$ row-wise entries for node i are all-zero and $O_{(r,s)}^{i,i} = 1$ otherwise if the bipartite association exists.This regularization aligns the representations of nodes that are connected by cross-edges to lie on the same space and be closer to each other.

3.3 Joint embedding with Consensus Regularization

Having obtained rich node representations that incorporated local, global and cross-layer structural information at every relational layer, we need a mechanism for aggregation of nodes' different relation-specific representations into a joint embedding. Since different layers may contribute differently to the end task, we use an attention mechanism to aggregate information across relations as,

$$\mathcal{J}_r^i = \frac{\exp(L_r \cdot U_r^i)}{\sum_{r'\in\mathcal{R}} \exp(L_{r'} \cdot U_r^i)}, \qquad U^i = \sum_{r\in\mathcal{R}} \mathcal{J}_r^i U_r^i \tag{8}$$

where L_r is the layer-specific embedding and $\mathcal{J}_r^i$ is the importance of layer r for node i. The importance score is computed by measuring the dot product similarity between the relational node embedding U_r^i and learned layer embedding L_r.

Additionally, to obtain a consensus embedding [20], we leverage the corrupted node representations $\tilde{U}_r : r \in \mathcal{R}$ that we computed for InfoMax optimization in Eqn: 1. A consensus node embedding $Z \in \mathbb{R}^{|\mathcal{V}|\times d}$ is learned with a regularization strategy that minimizes

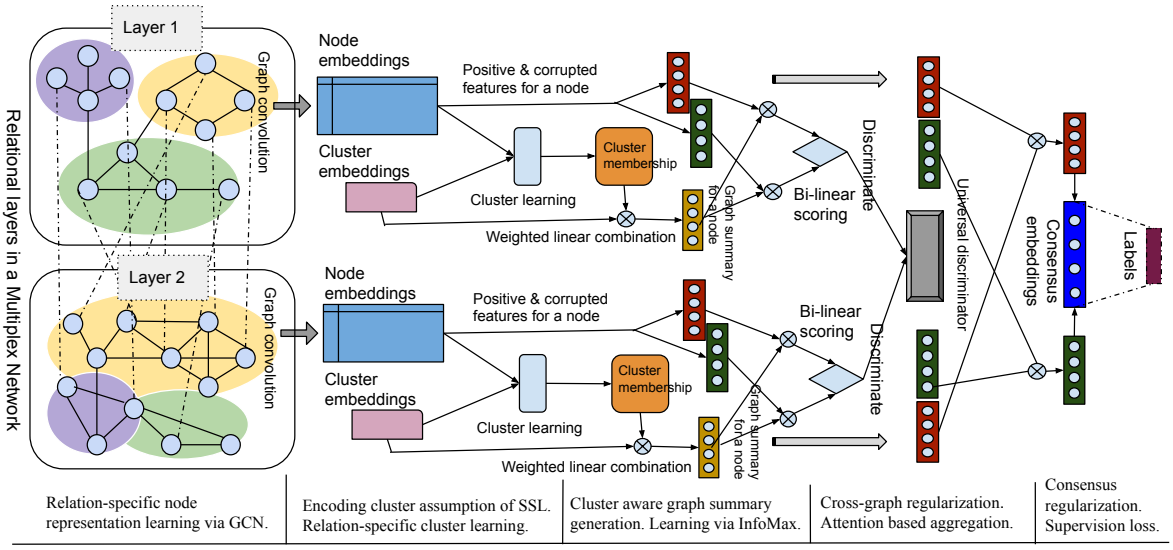


Figure 2: Semi-Supervised Deep Clustered Multiplex (SSDCM) with structure-aware graph summary generation
Explanation of used color-codes. Green: True node embeddings, Red: Corrupted node embeddings, Blue: Final consensus node embeddings. Orange: Learned cluster membership for nodes, Purple: Label information. The color-coded grouping of nodes in relational layers of the example multiplex network denotes different clusters.

the dis-agreement between Z and attention-weighted aggregated true node representations U, while maximizing the dis-agreement between combined corrupted node representations $\tilde{U}$ (re-using the same attention weights). The final consensus node embedding Z is generated as,

$$\mathfrak{O}_{Cons} = \|Z - U\|_F^2 - \|Z - \tilde{U}\|_F^2 \tag{9}$$

3.4 Semi-Supervised Deep Multiplex Clustered InfoMax

We predict labels $\hat{Y}$ for nodes using their consensus embeddings Z. We project Z into the label space using weights $W_Y \in \mathbb{R}^{d \times |\mathcal{Q}|}$ and normalize it with σ, a *softmax* or *sigmoid* activation function for multi-class and multi-label tasks respectively. The prediction function is learned by minimizing the following cross-entropy loss,

$$\mathfrak{O}_{Sup} = -\frac{1}{|\mathcal{L}|} \sum_{i \in \mathcal{L}} \sum_{q \in \mathcal{Q}} Y_{iq} \ln \hat{Y}_{iq} \tag{10}$$

$$\hat{Y} = \sigma(ZW_Y)$$

Finally, the overall semi-supervised learning process to obtain rich node representations that capture local-global structures in a multiplex network is obtained by jointly optimizing the equation below that optimizes different necessary components. We leverage hyper-parameters $\alpha, \beta, \gamma, \zeta, \theta$ to fine-tune the contributions of different terms.

$$\mathfrak{O} = \alpha * \mathfrak{O}_{MI} + \beta * \mathfrak{O}_{Cross} + \gamma * \mathfrak{O}_{Cons} + \zeta * \mathfrak{O}_{Clus} + \theta * \mathfrak{O}_{Sup} \tag{11}$$

Empirically, we find that our objective function is not very sensitive to variation in α, β values. Therefore, we fix their values as $\alpha = 1.0, \beta = 0.001$. Finally, we only tuned variables γ, ζ, θ in the above objective function to analyze the contributions of network, cluster, and label information. We discuss this further in Table 10 of Appendix A.3.1.

4 RELATED WORKS

Here, we discuss related representation learning literature focused on multiplex networks.

Network Representation Learning (NRL). Network Representation Learning (NRL) methods for multiplex networks use different learning paradigms such as matrix factorization [12]; random-walk based objectives [14, 33] and graph neural network architectures [4, 15, 20].

NRL models for multiplex networks have aimed at capturing different aspects of this multi-layered data. Modeling multi-layer data might require one to capture local [12, 17] and global network structures [15, 20] within each layer; leverage cross-layer edges [4, 12, 17, 19] between layers; encode node features [4, 20]; integrate information from multiple layers into a unified feature space [14, 33]; Optimize for single objective [24, 31] or jointly optimize for different objectives at different layers [4, 15].

Global context-based NRL. In general, random-walk-based methods and GCNs are limited to capturing k-hop local contexts of nodes only. While matrix factorization methods embed the entire graph, they are neither scalable nor powerful than the other two. Only a few studies capture the global structures into node embedding learning in both homogeneous and multiplex networks.

GUNets [3] introduced pooling and unpooling operations on multi-graphs for homogeneous networks. Its prominence-based node pooling captures the global graph structures and local graph structures using learnable projection vectors and GCNs. Deep Multi-Graph Clustering (DMGC) [15] is the first NRL study to learn global structures for multilayer networks explicitly. It proposes an attentive unsupervised mechanism to encode the cluster structures into multi-graphs based on a similarity-based cluster kernel.

InfoMax based NRL. *Deep Multiplex Graph Infomax (DMGI)*[20] employs the InfoMax principle for multiplex networks. It jointly maximizes the MI between the local and global graph patches across

the layers of a multiplex graph. It does so by learning a universal discriminator that discriminates positive and negative patch pairs across the relational layers. Simultaneously, it employs a regularization strategy that attentively aggregates the learned relation-specific node representations by reusing negative node representations used for learning the discriminator weights. HDGI [23] is a work similar to DMGI, aimed at heterogeneous networks. It adopts a semantic attention mechanism to aggregate metapath influenced node embeddings and discriminator-based learning strategy.

Semi-Supervised Learning (SSL). State-of-the-art methods MGCN [4], DMGI [20] are examples of SSL frameworks for multi-layer/ multiplex networks. MGCN proposes a layered graph convolution neural (GCN) architecture to preserve the within layer and cross-layer network structures by leveraging a cross-entropy loss function in each network layer. DMGI – though originally proposed as an unsupervised method, inculcates a semi-supervised variant that explicitly guides the learning of layer attention weights. We have HAN [31] and RGCN [24] from the domains of heterogeneous and knowledge graphs, respectively. HAN proposes a GNN architecture based on hierarchical node-level and metapath-level attention mechanisms. In contrast, RGCN proposes a graph convolution-based message passing framework facilitating effective weight sharing to avoid overfitting on rare relations.

In Table 1, we summarize competing methods in terms of important aspects of a multi-graph that they are designed to capture. These methods either lack strategies for 1) capturing *global structural information:* or 2) *aggregating* node information across different counterparts of the same node from different layers, 3) capturing useful *structures*. Even if there are global NRL methods like DMGI, DMGC, GUNets — they either use a naive mean-pooling approach for acquiring global graph representations or unsupervised clustering criteria/ importance pooling strategy to capture global graph structures that might not be useful given the end task is concerned. Our framework, SSDCM, differs in that we build upon a semi-supervised structure-aware version of *InfoMax* — which is first-of-its-kind to the best of our knowledge. Our objective is to learn global-structure enhanced node representations suitable for node-wise tasks capturing all aspects of multiplex graphs.

	Methods	Comparison						
		DMGC [15]	**DMGI** [20]	**HAN** [31]	**MGCN** [4]	**RGCN** [24]	**GUNets** [3]	**SSDCM**
Properties	attributes		✓	✓	✓		✓	✓
	within-network	✓	✓	✓	✓	✓	✓	✓
	cross-network	✓		–	✓	–	–	✓
	labels		✓	✓	✓	✓	✓	✓
	global structure	✓	✓				✓	✓
	aggregation		✓	✓	✓	✓		✓

* Dash marks denote Not Applicable (NA).

Table 1: Coverage of multiplex network features.

5 EXPERIMENTAL SETUP

Datasets. We evaluate our proposed algorithm SSDCM on a variety of datasets as mentioned in Table 2, from diverse domains, containing — both multi-class and multi-label datasets, as well as, attributed and non-attributed datasets. Refer to Table 9, Appendix A.3.2 for additional dataset details.

Dataset	Layers	Nodes	Edges(Total)	Features	Labels
ACM [31]	5	7427	24536689	767	5
DBLP [31]	4	4057	17976710	8920	4
SLAP [35]	6	20419	8207130	2695	15
IMDB-MC [20]	2	3550	80216	2000	3
IMDB-ML [21]	3	18352	2505797	1000	9
FLICKR [15]	2	10364	506051	–	7
AMAZON [20]	3	17857	2194389	2395	5

Table 2: Statistics of datasets **(Refer to Table 9, Appendix A.3.2 for details)**

Baselines. We chose State-Of-The-Art (SOTA) competing methods applicable to a diverse range of multi-graph settings. The compared methods can be roughly categorized into the following classes: multi-layered network-based embedding approaches — DMGC, MGCN; multiplex network embedding — DMGI; heterogeneous network embedding — HAN; multi-relational network embedding — RGCN; pooling method in multi-graph setting — GUNets.

Evaluation Strategy. We use a random sampling strategy to split the nodes into train, validation, and test set. We choose one-third of the labeled examples as train nodes. We keep the validation set size as half of the train set size. Thus, half of the total nodes are kept for evaluation purposes as test-set. Our experimental setup is summarized in Table 10, Appendix A.3.1. For the methods applicable to non-attributed graphs, namely, RGCN and DMGC – we implement attributed versions. For RGCN, we customized the relational GCN to take node features as input. For DMGC that uses relation-specific autoencoders to reconstruct the layer adjacencies, we input another array of feature-specific autoencoders. The feature-based autoencoders jointly learn a common hidden node representation along with the relational autoencoders and reconstruct layers' node features. To set up attributed NRL methods for FLICKR, we leverage the layer adjacencies as node features. We simply obtain the average of layer node embeddings as final representations for the methods with no specific node embedding aggregation strategy. We provide additional details of our analysis to facilitate replicability of results in Appendix A.3. We also provide our code [1].

6 RESULTS

We demonstrate the effectiveness of the proposed framework on four tasks, namely, node classification, node clustering, visualization, and similarity search. The details of the task-specific experiment setup, along with insights on results, are discussed below.

6.1 Node Classification

For semi-supervised methods, we use the predicted labels directly to compute the node classification scores based on ground-truth. We train a logistic regression classifier on the learned node embeddings of the training data for unsupervised methods and report the performance of the predictor on the test node embeddings averaged over twenty runs. We report the test-set performance that corresponds to the best validation-set performance for a fair comparison. Micro-F1 and Macro-F1 scores are reported as node classification results in Tables [3, 4] respectively. From Tables [3, 4], it is clear SSDCM

[1] https://github.com/anasuamitra/ssdcm

Micro-F1	ACM	DBLP	SLAP	FLICKR	AMAZON	IMDB-MC	IMDB-ML
DMGC	42.822	84.684	29.819	50.308	69.716	56.278	44.765
RGCN	39.118	83.514	26.914	82.69	72.957	62.542	49.802
GUNets	46.428	87.124	32.985	87.607	77.177	52.508	43.988
MGCN	52.458	87.003	29.563	91.307	84.083	63.384	48.059
HAN	77.441	85.989	30.976	89.478	83.77	62.353	47.117
DMGI	81.205	89.43	30.03	91.225	89.422	65.21	53.413
SSDCM	**88.324**	**94.988**	**33.597**	**96.261**	**92.195**	**67.796**	**54.055**

Table 3: Node classification results: Micro-F1 scores (%)

Macro-F1	ACM	DBLP	SLAP	FLICKR	AMAZON	IMDB-MC	IMDB-ML
DMGC	39.679	83.279	21.581	46.122	64.013	54.699	29.122
RGCN	38.665	82.86	24.119	81.47	68.323	62.17	45.285
GUNets	41.433	86.426	18.807	85.708	74.332	51.039	27.591
MGCN	46.853	85.462	25.717	91.07	82.349	62.876	38.821
HAN	78.009	85.154	25.413	89.174	82.344	61.891	35.181
DMGI	80.802	88.828	24.854	91.928	88.114	65.066	48.122
SSDCM	**88.571**	**94.681**	**28.072**	**96.147**	**91.973**	**67.803**	**51.756**

Table 4: Node classification results: Macro-F1 scores (%)

is the best performing model on all the datasets, by a significant margin. In comparison, DMGI gives the second-best performance on most datasets except on SLAP and FLICKR (for Micro-F1 scores).

6.2 Node Clustering

NMI-N	ACM	DBLP	SLAP	FLICKR	AMAZON	IMDB-MC	IMDB-ML
DMGC	0.421	0.532	0.245	0.488	0.468	0.185	0.076
RGCN	0.324	0.559	0.24	0.715	0.405	0.193	**0.102**
GUNets	0.65	0.742	0.251	0.758	0.519	0.108	0.036
MGCN	0.41	0.738	0.278	0.76	0.528	0.195	0.033
HAN	0.939	0.66	0.278	0.639	0.519	0.178	0.055
DMGI	0.837	0.682	0.275	0.644	0.568	0.194	0.056
SSDCM	**0.947**	**0.819**	**0.284**	**0.822**	**0.635**	**0.223**	0.085

Table 5: Node clustering results: NMI scores

We only cluster the test nodes to evaluate performance on the node clustering task. We give the test node embeddings to the clustering algorithm as input to predict the clusters. We run each experiment ten times and report the average scores in Table 5. K-Means and Fuzzy C-Means algorithms are used to predict clusters in multi-class and multi-label data, respectively. For multi-label data, we take the top q number of predicted clusters, where q is the number of classes that a node is associated with, to compare against the set of ground-truth clusters. We evaluate the obtained clusters against ground truth classes and report the Normalized Mutual Information (NMI) [16] scores. We use Overlapping NMI (ONMI) [11] for overlapping clusters to evaluate the multi-label datasets. Here we consider two kinds of clustering to demonstrate the effectiveness of our method. One is node clustering through clustering algorithms that takes final node embeddings as input. We refer to this clustering score as *NMI-N*. Another is directly predicting clusters from the cluster membership matrices learned during the optimization process and comparing it to the ground-truth to evaluate the clustering performance. The latter score, referred to as *NMI-C*, is only applicable to SSDCM and DMGC. From Table 5, we can see that except on IMDB-ML, SSDCM outperforms all the competing methods on the clustering task. It beats the second-best performing model by 0.037, across all datasets on average — a significant improvement.

6.3 t-SNE Visualizations

We also visualize the superior clusterability of SSDCM's learned node representations for the FLICKR and AMAZON dataset in Figure 3 using t-Distributed Stochastic Neighbor Embedding (t-SNE) [27] visualization. The color code indicates functional classes for respective datasets. We choose the node embeddings that gave the best performance in node classification scores for all the competing methods. We can see that all the semi-supervised methods yield interpretable visualizations indicating clear inter-class separation. Among them, SSDCM obtains compact well-separated small clusters of the same class labels, which appear to be visually better-separated than the rest of the methods. We see similar trends in visualization for other datasets also (not shown here).

6.4 Node Similarity Search

In a similar setup to [20], we calculate the cosine similarity scores of embeddings among all pairs of nodes. For a query node, the rest of the nodes are ranked based on the similarity scores. We then retrieve top $K = \{5, 10, 20, 50, 100\}$ nodes to determine the fraction of retrieved nodes with the same labels as the query node, averaged by K. For multi-label graphs, instead of exact label matching, we use the Jaccard similarity to determine the relevance of the query and target nodes' label set. We compute this similarity search score for all nodes as a query and report the average. The similarity search results get a significant boost under our framework since our encoding of the SSL clusters puts nodes with similar labels together in the same cluster. Whereas DMGC's clustering criterion, DMGI's global pooling, and GUNet's node importance based pooling criterion – do not demonstrate a similar benefit. From Tables [3, 4], we see for SLAP and two versions of IMDB movie networks the classification score of the competing methods are close. But in similarity search, we can differentiate SSDCM as the best performing model among all. DMGI is the second-best performing model in node similarity search, similar to the node classification results. GUNets and DMGC are seen to perform worse than the rest.

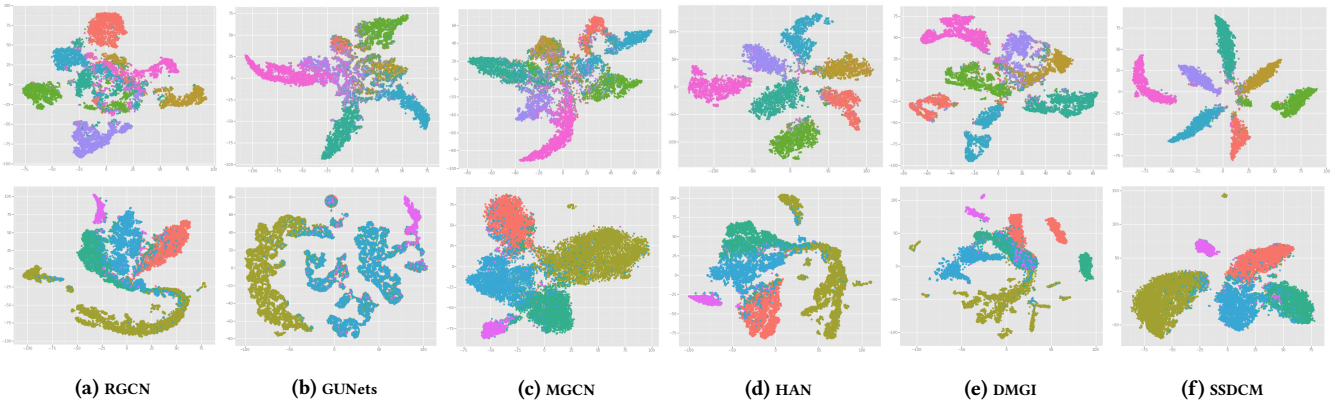
7 ANALYSIS

Herein we conduct an array of drill down experiments to shed light on the key components of our proposed SSDCM framework.

7.1 Novelty of cluster-based graph summary

Micro-F1 Scores	IMDB_MC	ACM	DBLP	AMAZON	FLICKR
SSDCM [global pool]	65.942	84.176	91.592	90.62	92.698
SSDCM [top-K pool]	63.908	84.218	90.683	90.34	93.714
SSDCM [SAG pool]	66.574	83.176	92.859	90.878	93.015
SSDCM [ASAP pool]	66.365	85.064	91.782	90.844	94.689
SSDCM [cluster aware graph summary]	**67.796**	**88.324**	**94.988**	**92.195**	**96.261**

Table 6: Novelty of cluster-based graph summary



(a) RGCN (b) GUNets (c) MGCN (d) HAN (e) DMGI (f) SSDCM

Figure 3: t-SNE Visualization of node embeddings on FLICKR (top), AMAZON (bottom) for all the SSL methods

Please refer to Section: 5 for the candidate methods for which the t-SNE visualizations are plotted here. The color codes indicate functional classes (FLICKR: 7, AMAZON: 5).

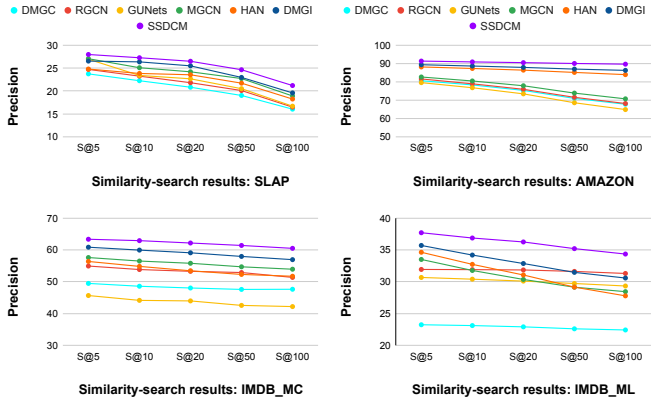


Figure 4: Comparing similarity search results

In Table 6, we delve deeper into how good the cluster-aware graph summary representation (Equation 3) is for the universal discriminator (Equation 1). We consider alternative SOTA pooling methods — Top-K [3], SAG [34] and ASAP [22] for generating graph summaries in the SSDCM framework. Top-K pool realizes node importance based pooling strategy via learning a projection vector. In comparison, the SAG pool improves upon the former by encoding structural information from graphs using GNNs. Adaptive Structure Aware Pooling (ASAP) is a new SOTA method that considers the cluster structures from graphs. It proposes a self-attentive GCN architecture *Master2Token* to learn clusters and uses a cluster fitness-based scoring strategy to pool underlying graph structures in phases [2] These pooling strategies generate a common graph summary for the whole graph, which is fed to the discriminator along with the node embeddings. On the contrary, our cluster-aware graph summary has a node's perspective, i.e., the global graph summaries vary from node to node based on its associated cluster structures. For the nodes that share membership under a common set of clusters, the structure-aware graph summaries are similar. That makes the universal discriminator more powerful for discriminating the local and global patch pair representations from the false pairs across the relations.

Here, we keep SSDCM's cluster learning component intact and use various pooling strategies to train the discriminator. The discriminator, thus, does not have any relation to the learned clusters and uses a common global summary paired with each node. In Table 6, we see, structure-aware pooling methods are beneficial. *We see that the pooling variants with SSDCM have better performance than DMGI's best-reported performance (Table 3), mainly due to learning of the clusters and using advanced pooling strategies* in place of DMGI's mean-pooling. Empirically, we observe that SSDCM with the alternative pooling variants struggle to converge consistently. However, SSDCM with a cluster-based graph summary does not suffer from similar convergence issues. *Our proposed architecture in the last row outperforms all the candidate pooling techniques significantly on every dataset, depicting the effectiveness of our cluster-aware graph summary representations.*

7.2 Effect of various regularizations

Comparison	IMDB-MC		FLICKR		ACM	
	Micro-F1	NMI-N	Micro-F1	NMI-N	Micro-F1	NMI-N
SSDCM	**67.796**	**0.22325**	**96.261**	**0.82171**	**88.324**	**0.94650**
SSDCM−cross	66.613	0.20451	94.182	0.79671	86.371	0.91611
SSDCM−cons	66.069	0.20138	91.836	0.73658	84.889	0.88139
SSDCM−(cons+cross)	64.971	0.18735	88.374	0.71629	83.961	0.85420

Table 7: Effect of cross and consensus regularizations

'+' and '-' sign denote augmentation or elimination of the components followed.

Here we compare the results of SSDCM without cross regularization with SSDCM to understand the influence of this factor. *We see that removing cross-edge based regularization from the layer-wise node embeddings degrades the performance of the SSDCM considerably,* especially on FLICKR and ACM. Next, we verify the usefulness of learning a final consensus node embedding from the attention-aggregated positive and corrupted node embeddings. Recall that our universal discriminator learns to discriminate between true

[2]We use the Pytorch Geometric [2] library for candidate pooling methods.

local–global patches from the corrupted ones with the intuition that the corrupted embeddings seek to improve the discriminative power of the resulting embeddings. *We see that the consensus regularization indeed plays an essential role in enriching the final node embeddings – an observation similar to DMGI's.* From Table 7, we see that the consensus embeddings improve the performance of Micro-F1 scores by a maximum of 4.425% on FLICKR, followed by 3.435%, 1.558% improvements on ACM, IMDB-MC respectively. *SSDCM–(cons+cross) gives the worst performance among all the compared variations.* The reasons behind this are self-explanatory – a) no cross-edges to align the relational node representations to each-other, b) it lacks in a discriminative capacity.

8 CONCLUSION

In this study, we propose a semi-supervised framework for representation learning in multiplex networks. This framework incorporates a unique InfoMax based learning strategy to maximize the MI between local and contextualized global graph summaries for effective joint modeling of nodes and clusters. Further, we use the cross-layer links to impose further regularization of the embeddings across the various layers of the multiplex graph. Our novel approach, dubbed SSDCM, improves over the state-of-the-art over a wide range of experimental settings and four distinct downstream tasks, namely, classification, clustering, visualization, and similarity search, demonstrating the proposed framework's overall effectiveness. In the future, we hope to extend this work in a couple of ways. First, we hope to improve the scalability of the approach – perhaps by leveraging a graph coarsening and refinement strategy[13] within SSDCM. Second, we propose to see if the ideas we have presented can be generalized for other types of multi-layer graphs (i.e., not just multiplex networks).

ACKNOWLEDGMENTS

The authors of IIT Guwahati acknowledge the Department of Biotechnology, Government of India for the financial support for the Project BT/COE/34/SP28408/2018. S. Parthasarathy acknowledges NSF grants: SES-1949037, OAC-2018627, CCF-2028944, and NIH grant: R01 HD088545-01A. B. Ravindran acknowledges partial support of a grant from Intel Technology India Pvt. Ltd. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the funding agencies.

REFERENCES

[1] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeswar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and R Devon Hjelm. 2018. Mine: mutual information neural estimation. In *International Conference on Learning Representations*.
[2] Matthias Fey and Jan E. Lenssen. 2019. Fast Graph Representation Learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs*.
[3] Hongyang Gao and Shuiwang Ji. 2019. Graph u-nets. In *international conference on machine learning*. PMLR, 2083–2092.
[4] Mahsa Ghorbani, Mahdieh Soleymani Baghshah, and Hamid R Rabiee. 2019. MGCN: semi-supervised classification in multi-layer graphs with graph convolutional networks. In *Proceedings of the 2019 ASONAM*. 208–211.
[5] Roger Guimera and Luis A Nunes Amaral. 2005. Functional cartography of complex metabolic networks. *nature* 433, 7028 (2005), 895–900.
[6] W. L. Hamilton. 2020. *Graph Representation Learning*. Morgan and Claypool.
[7] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *WWW*. 507–517.
[8] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).
[9] Mikko Kivelä, Alex Arenas, Marc Barthelemy, James P Gleeson, Yamir Moreno, and Mason A Porter. 2014. Multilayer networks. *Journal of complex networks* 2, 3 (2014), 203–271.
[10] Xiangnan Kong, Philip S Yu, Ying Ding, and David J Wild. 2012. Meta path-based collective classification in heterogeneous information networks. In *CIKM*. 1567–1571.
[11] Andrea Lancichinetti, Santo Fortunato, and János Kertész. 2009. Detecting the overlapping and hierarchical community structure in complex networks. *New journal of physics* 11, 3 (2009), 033015.
[12] Jundong Li, Chen Chen, Hanghang Tong, and Huan Liu. 2018. Multi-layered network embedding. In *SDM*. SIAM, 684–692.
[13] Jiongqian Liang, Saket Gurukar, and Srinivasan Parthasarathy. 2021. MILE: A Multi-Level Framework for Scalable Graph Embedding. *in ICWSM* (2021).
[14] Weiyi Liu, Pin-Yu Chen, Sailung Yeung, Toyotaro Suzumura, and Lingli Chen. 2017. Principled multilayer network embedding. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE, 134–141.
[15] Dongsheng Luo, Jingchao Ni, Suhang Wang, Yuchen Bian, Xiong Yu, and Xiang Zhang. 2020. Deep Multi-Graph Clustering via Attentive Cross-Graph Association. In *WSDM*. 393–401.
[16] Christopher D Manning, Hinrich Schütze, and Prabhakar Raghavan. 2008. *Introduction to information retrieval*. Cambridge university press.
[17] Ryuta Matsuno and Tsuyoshi Murata. 2018. MELL: effective embedding method for multiplex networks. In *Companion Proceedings of the The Web Conference 2018*. 1261–1268.
[18] Anasua Mitra, Priyesh Vijayan, Srinivasan Parthasarathy, and Balaraman Ravindran. 2020. A Unified Non-Negative Matrix Factorization Framework for Semi Supervised Learning on Graphs. In *Proceedings of the 2020 SIAM International Conference on Data Mining*. SIAM, 487–495.
[19] Jingchao Ni, Shiyu Chang, Xiao Liu, Wei Cheng, Haifeng Chen, Dongkuan Xu, and Xiang Zhang. 2018. Co-regularized deep multi-network embedding. In *WWW*. 469–478.
[20] Chanyoung Park, Donghyun Kim, Jiawei Han, and Hwanjo Yu. 2020. Unsupervised Attributed Multiplex Network Embedding.. In *AAAI*. 5371–5378.
[21] Trang Pham, Truyen Tran, Dinh Phung, and Svetha Venkatesh. 2017. Column networks for collective classification. In *AAAI*.
[22] Ekagra Ranjan, Soumya Sanyal, and Partha P Talukdar. 2020. ASAP: Adaptive Structure Aware Pooling for Learning Hierarchical Graph Representations.. In *AAAI*. 5470–5477.
[23] Yuxiang Ren, Bo Liu, Chao Huang, Peng Dai, Liefeng Bo, and Jiawei Zhang. 2020. HDGI: An Unsupervised Graph Neural Network for Representation Learning in Heterogeneous Graph. (2020).
[24] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *ESWC*. Springer, 593–607.
[25] Fan-Yun Sun, Jordan Hoffman, Vikas Verma, and Jian Tang. 2019. InfoGraph: Unsupervised and Semi-supervised Graph-Level Representation Learning via Mutual Information Maximization. In *International Conference on Learning Representations*.
[26] Fei Tian, Bin Gao, Qing Cui, Enhong Chen, and Tie-Yan Liu. 2014. Learning deep representations for graph clustering. In *AAAI*, Vol. 28.
[27] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *JMLR* 9, 11 (2008).
[28] Petar Veličković, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2019. Deep graph infomax. In *International Conference on Learning Representations*.
[29] Daixin Wang, Peng Cui, and Wenwu Zhu. 2016. Structural deep network embedding. In *ACM SIGKDD*. 1225–1234.
[30] Xiao Wang, Peng Cui, Jing Wang, Jian Pei, Wenwu Zhu, and Shiqiang Yang. 2017. Community preserving network embedding.. In *AAAI*, Vol. 17. 203–209.
[31] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous graph attention network. In *WWW*. 2022–2032.
[32] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems* (2020).
[33] Hongming Zhang, Liwei Qiu, Lingling Yi, and Yangqiu Song. 2018. Scalable Multiplex Network Embedding.. In *IJCAI*, Vol. 18. 3082–3088.
[34] Liang Zhang, Xudong Wang, Hongsheng Li, Guangming Zhu, Peiyi Shen, Ping Li, Xiaoyuan Lu, Syed Afaq Ali Shah, and Mohammed Bennamoun. 2020. Structure-Feature based Graph Self-adaptive Pooling. In *WWW*. 3098–3104.
[35] Yizhou Zhang, Yun Xiong, Xiangnan Kong, Shanshan Li, Jinhong Mi, and Yangyong Zhu. 2018. Deep collective classification in heterogeneous information networks. In *WWW*. 399–408.
[36] Marinka Zitnik, Monica Agrawal, and Jure Leskovec. 2018. Modeling polypharmacy side effects with graph convolutional networks. *Bioinformatics* 34, 13 (2018), i457–i466.

A APPENDIX

A.1 Ablation study

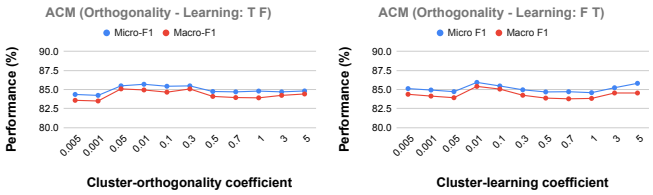


Figure 5: Ablation study of cluster learning components

Variants	Micro F1	Macro F1	NMI-N	NMI-C
SSDCM (OL : FT)	85.584	84.83	0.889	0.518
SSDCM (OL : TF)	84.795	83.907	0.868	0.391
SSDCM (OL : TT)	**88.324**	**88.571**	**0.947**	**0.651**

Symbol meanings – A: cluster assignment, L: cluster learning, O: cluster orthogonality. T: True, F: False – denotes absence or presence of respective terms.

Table 8: Impact of various cluster learning components

In Table 8, we study the impact of cluster related terms on the end-task performances by removing the relevant terms in two binary combinations. In OL: FT and OL: TF configurations, we remove the cluster orthogonality term and the semi-supervised cluster learning term, respectively. Removing the cluster learning term significantly impacts the NMI N and C scores by reducing the performance by 0.079 and 0.26 points. This configuration moderately affects the F1 scores. Removing the orthogonality term affects the classification performances with 2.74%, 3.771% reductions in Micro and Macro F1 scores. These reductions are less than the performance drops gotten from removing the cluster learning term in the case of F1 scores but still play a significant role. *The cluster learning term is seen to be more useful than the cluster orthogonality term for learning the cluster membership matrix.*

In Figure 5, we consider two possible combinations, namely, cluster assignment (Eqn 4)–learning (Eqn 6)–orthogonality (Eqn 5) as LO: FT and TF (T: True, F: False), for dissecting the cluster learning objective. We perform a range search to see under which settings the best classification performances is achieved by varying a particular cluster related term in a range while removing or keeping the rest of the terms intact. The terms are varied in range of $\in \{0.005, 0.001, 0.05, 0.01, 0.1, 0.3, 0.5, 0.7, 1, 3, 5\}$. In FT configuration, cluster orthogonality is varied in the absence of the cluster learning term. It gives best performance for values $\in \{0.05, 0.3\}$. Over a higher range of values, the performances become less fluctuating. In TF configuration, cluster learning is varied in the absence of cluster orthogonality. At 0.01 it gives the best performance in terms of Micro and Macro F1 for ACM. Again, an upward trend in performances can be seen for values $\in [1-5]$.

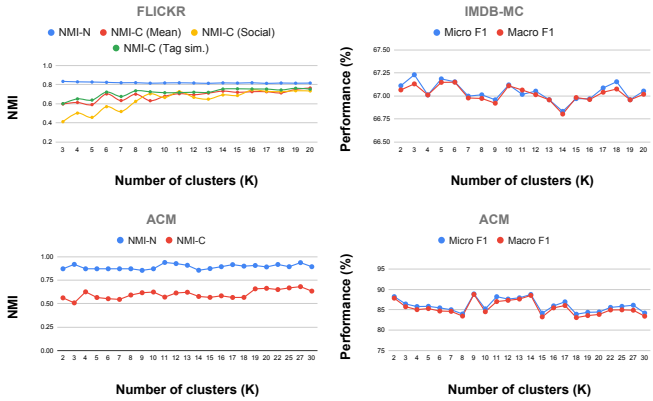


Figure 6: Varying number of clusters

Number of clusters K is varied for FLICKR, IMDB-MC and ACM. i) Micro, Macro F1 scores (on right), ii) NMI using node embeddings and cluster memberships (on left) are plotted. Best performances of DMGI (no cluster learning) are – a) for FLICKR, NMI-N: 0.644, **b) for IMDB-MC, Micro-F1:** 65.210, **Macro-F1:** 65.066, **and c) for ACM, Micro-F1:** 81.205, **Macro-F1:** 80.802, **NMI-N:** 0.837

A.2 Varying number of clusters

Here we study SSDCM's sensitivity towards varying the number of clusters K. We also verify whether there is a need to learn the cluster structures at all. We take the optimal hyper-parameter combination and vary the number of clusters in the range $[2-20]$ and $[2-30]$ for FLICKR, IMDB-MC, ACM, respectively. *Compared to DMGI's best performance scores, clear differences can be seen in Figure 6 for SSDCM that speaks to the effectiveness of learning clusters to enrich node embeddings.*

We plot the NMI-N and NMI-C scores (mean and layer-wise cluster memberships) while varying K for FLICKR. We see less perturbation in NMI-N scores than NMI-C scores here. As K goes higher, the layer-wise and mean cluster membership based NMI scores increase before flattening at $K = 20$. For IMDB-MC, We can see Micro F1 scores are best at $K = Q = 3$, i.e., when the number of classes and clusters are the same. For ACM, varying K improves Micro and Macro F1 scores at $K \in \{2, 3\} < (Q = 5)$, i.e., when SSDCM learns high-level clusters. Even when $K \geq Q$, i.e., when SSDCM learns small clusters of same class data. We see a gradual improvement in both the NMI scores for ACM when $k \in [9-30]$. NMI-N and NMI-C tend to give different NMI scores. The possible interpretation of this performance difference lies in the fact that – in NMI-N, the K-Means algorithm applied to the node embeddings of considerable hidden dimensions ($d = 64$) and the NMI scores are calculated for ground truth clusters. Whereas, cluster memberships H are of comparatively low dimensions (K), and in NMI-C, we directly use the learned cluster membership probabilities to derive the NMI scores.

A.3 Reproducibility

A.3.1 Baselines. In Table 10, we give the details of hyper-parameter range search for all the competing methods — which is self-explanatory.

	Nodes	Layers	Node Types	Intra-Layer Relations	Edges	Features	Weighted	Directed	Multi-Class	Classes
ACM [31]	7427	5	**Paper (P)** Author (A) Proceeding (V) Institute (I) Subject (S)	PAP PAIAP PSP PVP PP	118453 8353678 14997105 1048129 19324	767 Paper Title & Abstract	True	False	True	5 Conference (C)
DBLP [31]	4057	4	**Author (A)** Paper (P) Conference (C) Term (T)	APA APAPA APCPA APTPA	11113 40703 5000495 12924399	8920 Paper Title & Abstract	True	False	True	4 Research Field (F)
SLAP [35]	20419	6	**Gene (G)** Gene Ontology (O) Pathway (P) Compound (C) Tissue (T) Disease (D)	GPG GTG GDCDG GOG GDG GG	832924 606974 36190 6371558 14988 344496	2695 Gene Ontology Description	True	False	True	15 Gene Family (F)
IMDB-MC [20]	3550	2	**Movie (M)** Actor (A) Director (D)	MAM MDM	66428 13788	2000 Movie Plot & Summary	True	False	True	3 Movie Genre (G)
IMDB-ML [21]	18352	3	**Movie (M)** Actor (A) Director (D) Actress (E)	MAM MDM MEM	1455381 923173 127243	1000 Movie Plot & Summary	True	True	False	9 Movie Genre (G)
FLICKR [15]	10364	2	**User (U)**	Friendship Tag-similarity	390938 115113	NA	True	True	True	7 Social Group (G)
AMAZON [20]	17857	3	**Product (P)**	Co-purchase Co-view Similar	1501401 590961 102027	2395 Product Description	True	False	True	5 Product Category (C)

Table 9: Dataset statistics

Methods	Experiment setup & hyper-parameter range
HAN [31]	l2 coefficient={0.0001, 0.0005, 0.001, 0.005}, learning rate={0.0001, 0.0005, 0.001, 0.005}, attention heads={1,2,4,8}, metapath attention dimension=128
MGCN [4]	network & label coefficient={0.01, 0.1, 1.0, 10.0}, l2 coefficient={0.0005, 0.005}, learning rate={0.0005, 0.001, 0.05, 0.01}, stacked GCNs=2
RGCN [24]	l2 coefficient={0.0005, 0.005}, learning rate={0.0005, 0.001, 0.05, 0.01}, no of bases=no of relations, number of hidden layers=2
GUNets [3]	l2 coefficient={0.0001, 0.001}, learning rate={0.01, 0.05, 0.001, 0.0005}, depth={3, 4, 5}, pool ratio={0.2, 0.4, 0.6, 0.8}
DMGC [15]	network coefficient={1.0, 0.8, 0.6, 0.4}, cross reg.={0.2, 0.4, 0.6, 0.8}, l2 coef=0.0001, learning rate={0.0005, 0.001, 0.05, 0.01}, stacks in AutoEncoder=2
DMGI [20]	network & label coefficient={0.001, 0.01, 0.1, 1.0}, l2 coefficient={0.0001, 0.001}, learning rate={0.0001, 0.0005, 0.001, 0.005}
SSDCM	network, label & cluster coefficient={0.001, 0.01, 0.1}, l2 coefficient=0.0001, cross regularization=0.001, learning rate={0.0001, 0.0005, 0.001, 0.005}
Default to All	hidden units=64, epochs=10000, patience=20, attention heads=2, non-linearity=prelu, no of clusters=no of classes, ϵ = 3.0, GCN layers = 2, validation set based hyperparameter tuning, features=adjacency for non-attributed graphs, no node aggregation strategy=mean-pooling.

Table 10: Experiment setup & hyper-parameter range search for competing methods

A.3.2 Datasets. Here in Table 9, we provide the detailed statistics of the datasets used for evaluation. We have used two versions of the IMDB dataset, one multi-class version **IMDB-MC** as used in DMGI, and, another multi-label version **IMDB-ML** from the Column Networks (CLN) [21]. In both versions, movie features are extracted from movie plot summary with movie genres as functional classes. We used multiplex versions of bibliographic datasets ACM and DBLP. For **ACM** [31], we extracted papers of five conferences [3] and created a multiplex network that includes layers of paper nodes connected by co-authors, similar subjects, similar venues, co-authors belonging from the same institutes, and citation relationships. Here, the task is to classify them according to the conferences as they are published. **DBLP** [31] is a multiplex network of authors. The authors are classified by their field of research-interests [4]. In both the bibliographic datasets, the terms extracted from the paper title and abstract are used as local features for the nodes under consideration. In **SLAP** [10, 35], multiple layers of interactions characterize a gene — including tissue-specific, biological pathways involved, disease associations, phylogenetic profile, gene expression, chemicals involved to treat associated diseases, etc. Each gene has ontology related terms associated with it as attributes, and it can belong to any of the most frequently occurring fifteen gene Families (F). We have **AMAZON** [7, 20], which is originally multiplex in nature, i.e., the multiplexity is not inferred from composite relations. This network is extracted from the product review metadata of the Amazon website. Target instances, i.e., products exhibit also-bought, also-viewed, and similar-to – three layers of relations among them. Most frequently occurred terms are extracted from product title as node features. The task is to classify the products into any product categories [5]. **FLICKR** [15] is a non-attributed multiplex social network of users (U) who belong to various communities of interest. It has a friendship layer and a tag similarity-based connection layer among the users. A user is categorized based on their membership to any of the social groups. In all the datasets mentioned in Table 9, cross-layer edges link two nodes in different layers if they refer to the same node.

[3]Conferences = ['KDD', 'WWW', 'SIGIR', 'SIGMOD', 'CIKM']

[4]Fields = [Data Mining (DM), Artificial Intelligence (AI), Computer Vision (CV), Natural language Processing (NLP)].

[5]Product Categories in AMAZON Multiplex Network = ['Appliances', 'Automotive', 'Patio Lawn & Garden', 'Pet Supplies', 'Home & Kitchen']