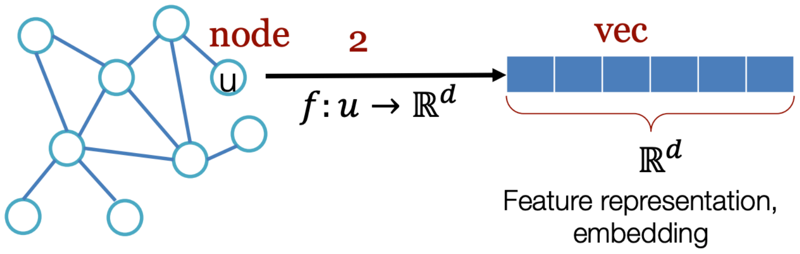
Unsupervised Attributed Multiplex Network Embedding

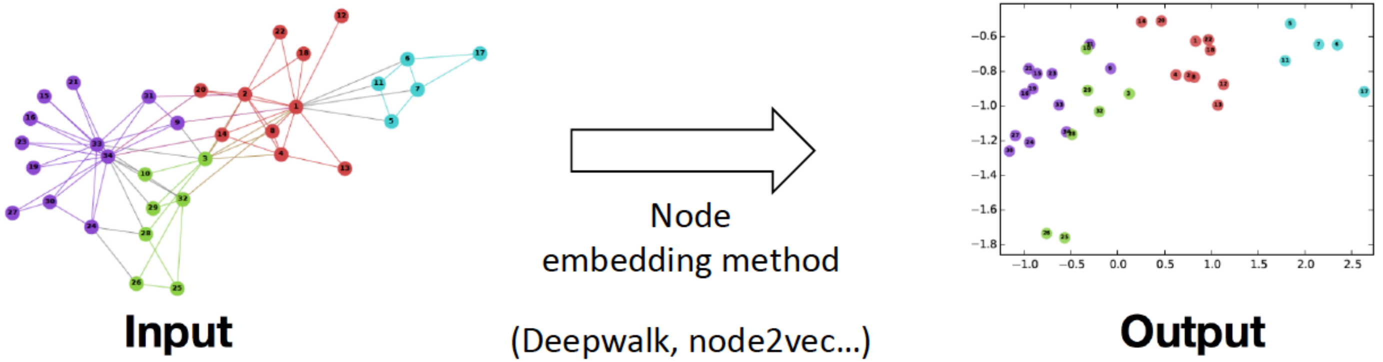
AAAI 2020

Presenter: Chanyoung Park

Network Embedding

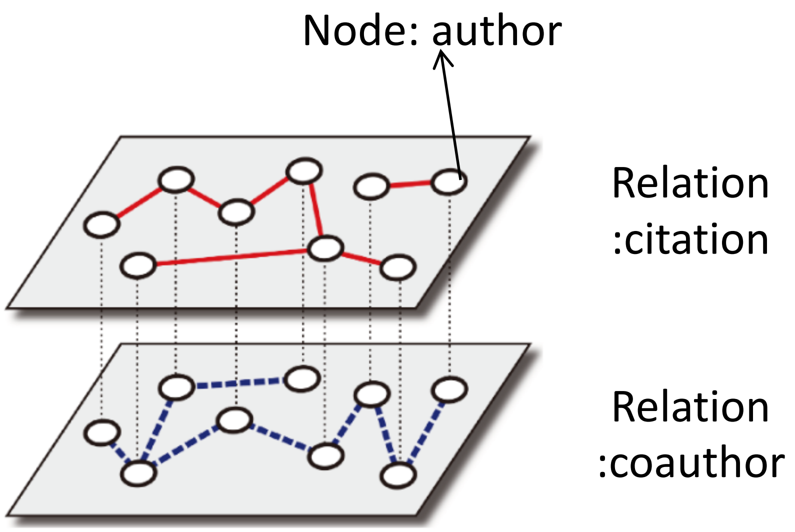
- Find a low-dimensional vector representation of each node in a graph while preserving t he network structure
  - Intuition: Similar nodes in a graph have similar vector representations





Multiplex Network (Multi-view network)

- A single node type, multiple edge types
  - **Example 1: Publication network**
    - Relationship between papers
      - Citation, share authors, share keywords
    - Relationship between authors
      - Co-author, co-advisor, co-citation
  - **Example 2: Movie database**
    - Relationship between movies
      - Common director, common actor
  - **Example 3: Social network**
    - Relationship between users
      - Family member, school friend, co-worker
  - **Example 4: E-commerce**
    - Relationship between items
      - Also-viewed, also-bought, bought-together
  - ...



Motivation

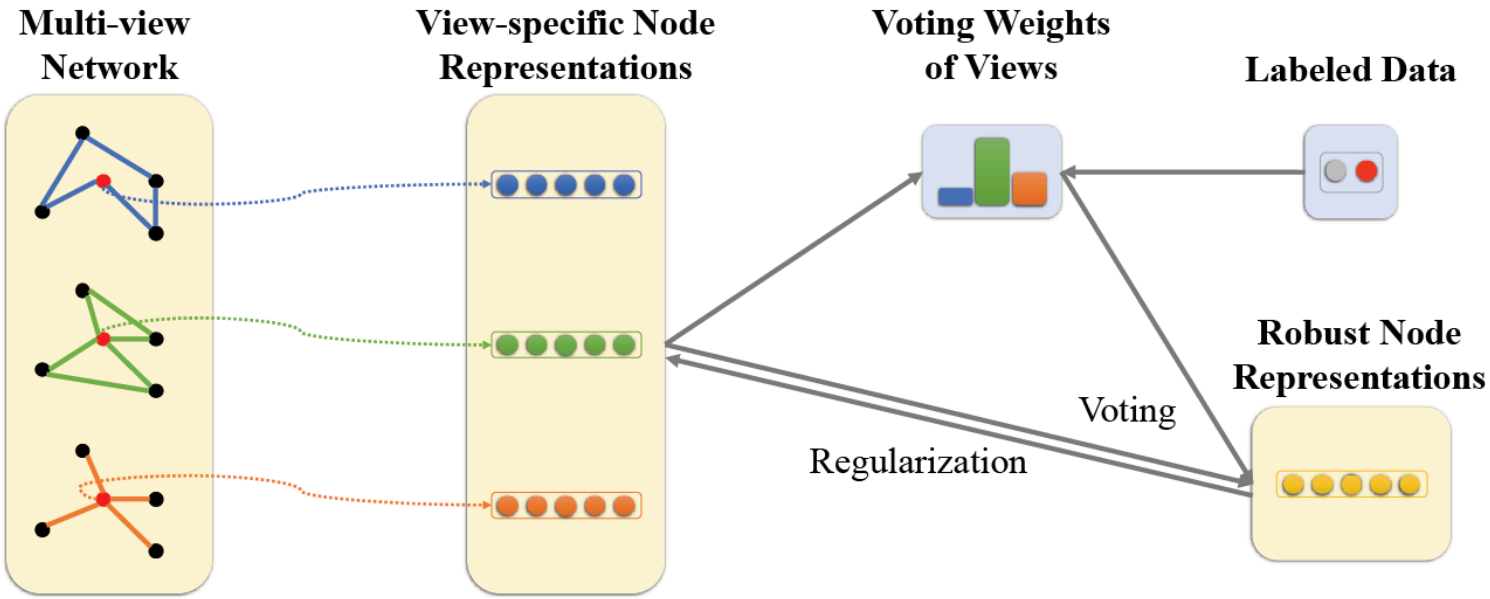
- Although different types of relations can independently form different graphs, **these gra phs are related**
  - Mutually help each other on various downstream tasks
- Example: Publication network
  - Inferring the topic of a paper only from its citations is difficult
  - But, also knowing other papers written by the same authors will help predict its topic
    - Authors usually work on a specific research topic
  - Furthermore, if **node attributes** are given, it becomes even easier
    - e.g., Abstract of the papers

This work

- Goal: Learn node representations in multiplex networks
  - Capture the interactions between multiple relation types
  - Consider node attributes if they are given
- Apply the learned representations for various downstream tasks
  - Node classification, node clustering, similarity search

Network Structure-based Methods

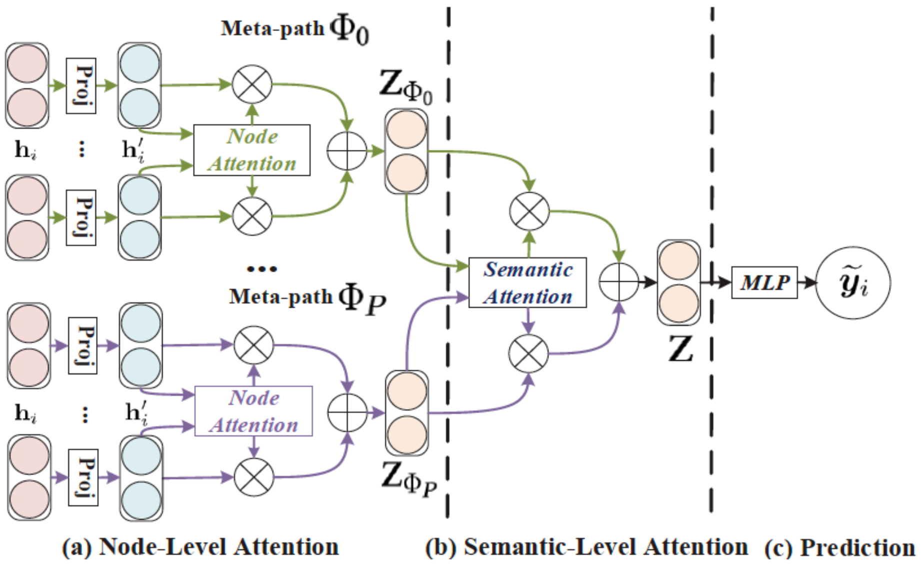
- Qu et al, 2017, Zhang et al, 2018, Chu et al, 2019

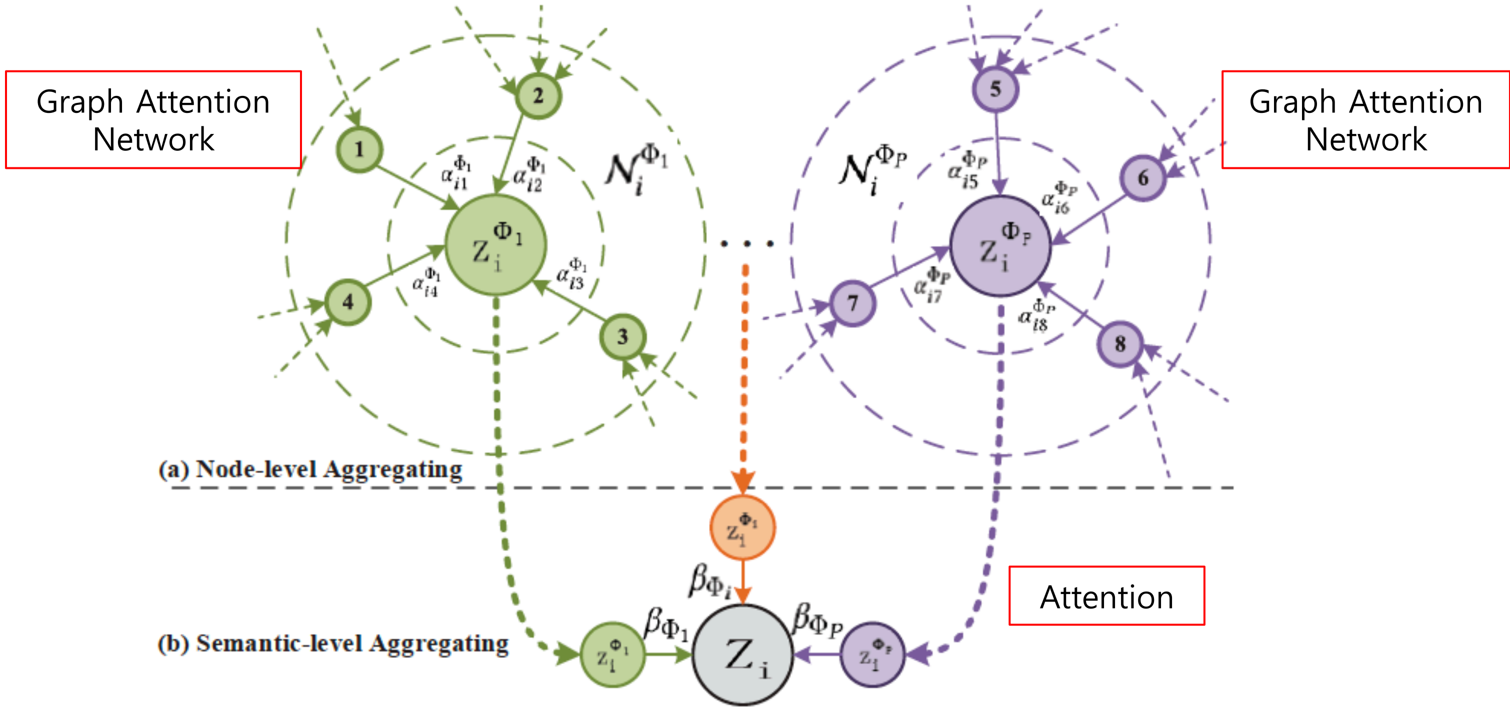


Do not consider node attributes
+ depend on label information

GNN-based methods

- Apply GCN (Ma et al, 2019) or GAT (Wang et al, 2019) to a multiplex network

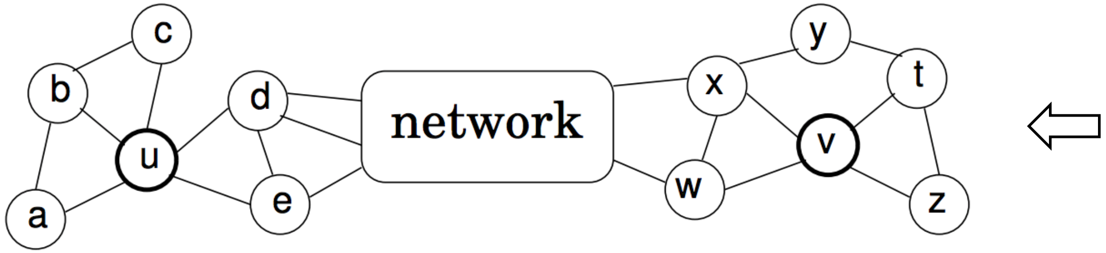




Do not consider the global information (neighborhood aggregation) + depend on label information

Limitations of Previous Work

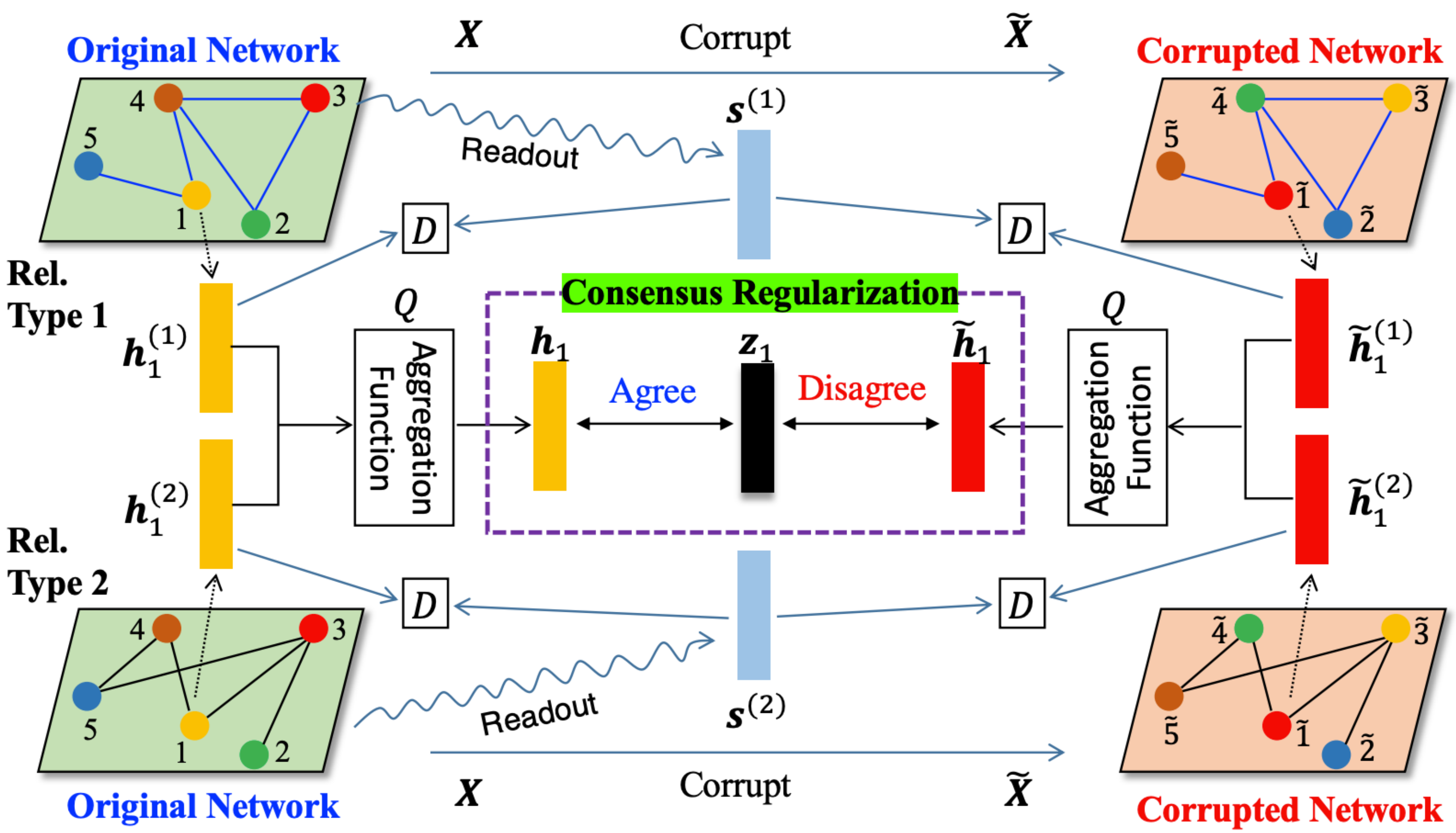
1. Ignore node attributes information
2. Rely on node labels
   - However, labels are not always given in the real world
3. Cannot capture the global property



u and v should have similar embeddings because they share similar structures

u and v are globally similar

Proposed Framework
: Deep Multiplex Graph Infomax (DMGI)



Background:
Mutual Information (MI)

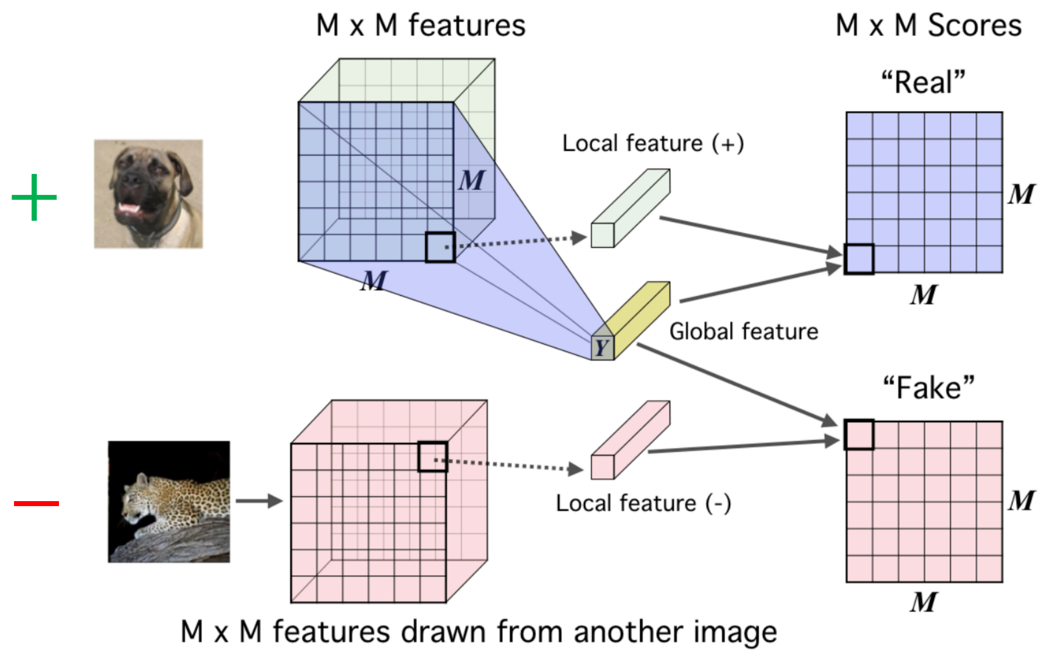
- Measures the amount of information that two variables share
- If X and Y are independent, then $P_{XY} = P_X P_Y$ → in this case, MI = 0

$$\begin{aligned}\mathcal{I}(X;Y) &= \mathbb{E}_{P_{XY}}\left[\log \frac{P_{XY}}{P_X P_Y}\right] \\ &= D_{\mathrm{KL}}(P_{XY} || P_X P_Y)\end{aligned}$$

- High MI? → One variable is always indicative of the other variable
- Recently, scalable estimation of mutual information was made both possible and practic al through **Mutual Information Neural Estimation** (MINE) [ICML18]

Deep Infomax (Hjelm et al, 2019)

- Unsupervised representation learning method for image data
- Intuition: **Maximize mutual information (MI)** between local patches and the global repr esentation of an image



Deep Infomax (Hjelm et al, 2019)

Discriminator tries to discriminate between "Real" and "Fake"

Deep Graph Infomax (Velickovic et al, 2019)

- Deep Graph Infomax (DGI) applies Deep Infomax on graph domain
- Unsupervised graph representation learning method that considers node features
- Notations

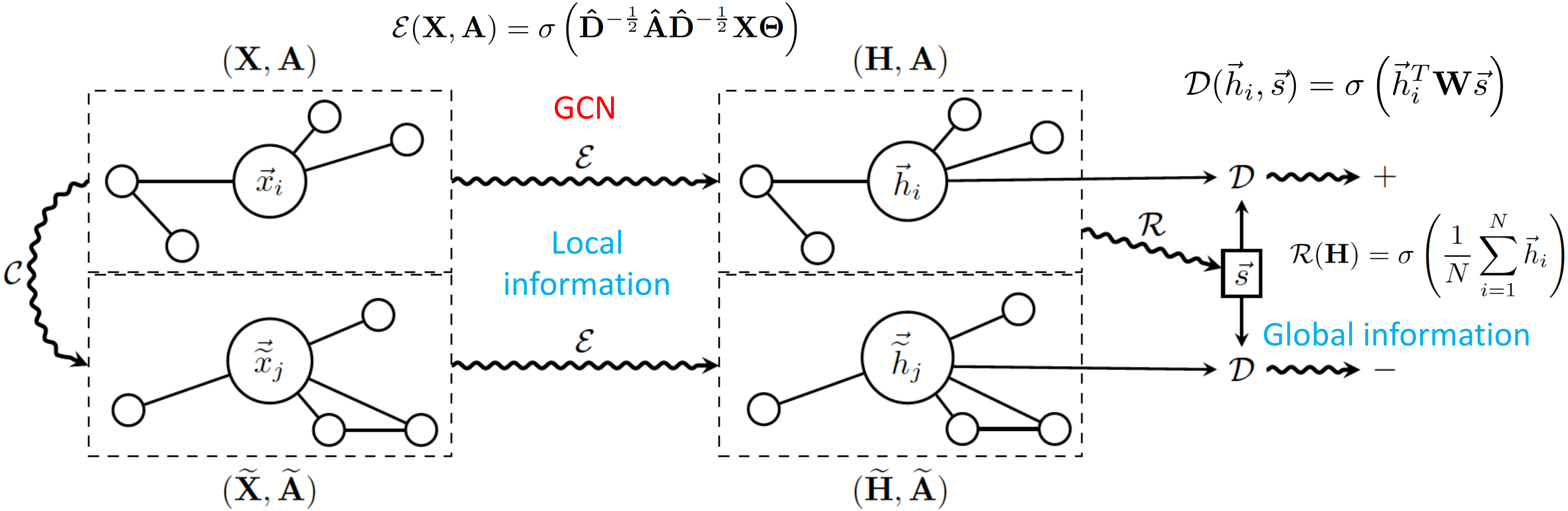
$\mathbf{X} = \{\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_N\}$: A set of node features (N: number of nodes) $\vec{x}_i \in \mathbb{R}^F$

$\mathbf{A} \in \mathbb{R}^{N \times N}$: Adjacency matrix

- Learn a **graph convolutional** encoder $\mathcal{E}(\mathbf{X}, \mathbf{A}) = \mathbf{H} = \{\vec{h}_1, \vec{h}_2, \ldots, \vec{h}_N\}$ $\vec{h}_i \in \mathbb{R}^{F'}$
  - Generates node representations by **repeated aggregation over local node neighborhoods**
  - $\vec{h}_i$ summarizes a patch of the graph centered around node i ($\approx$ patch representation)

Analogy: Local patch representation in an image == Node representation in a graph

Deep Graph Infomax (Velickovic et al, 2019)



$$\mathcal{L} = \frac{1}{N+M}\left(\sum_{i=1}^{N}\mathbb{E}_{(\mathbf{X},\mathbf{A})}\left[\log\mathcal{D}\left(\vec{h}_i,\vec{s}\right)\right] + \sum_{j=1}^{M}\mathbb{E}_{(\widetilde{\mathbf{X}},\widetilde{\mathbf{A}})}\left[\log\left(1-\mathcal{D}\left(\vec{\widetilde{h}}_j,\vec{s}\right)\right)\right]\right)$$

Maximizes the mutual information between the local patches and the graph-level global representation

Why Deep Graph Infomax?

- Considers node attributes
  - Graph convolution network based
- Does not rely on node labels
  - Unsupervised learning
- Captures global information
  - Mutual information maximizataion

How can we apply DGI on attributed multiplex networks?

Relation-specific Node Embedding

- Obtain relation specific node embedding for every node

$$g_r(\mathbf{X}, \mathbf{A}^{(r)} | \mathbf{W}^{(r)}) = \mathbf{H}^{(r)} = \sigma\left(\hat{\mathbf{D}}_r^{-\frac{1}{2}} \hat{\mathbf{A}}^{(r)} \hat{\mathbf{D}}_r^{-\frac{1}{2}} \mathbf{X}\mathbf{W}^{(r)}\right) \quad \hat{\mathbf{A}}^{(r)} = \mathbf{A}^{(r)} + w\mathbf{I}_n,\ \hat{D}_{ii} = \sum_j \hat{A}_{ij}$$

GCN

$\boldsymbol{A}^{(r)}$: Adjacency matrix w.r.t. relation r

$\boldsymbol{H}^{(r)}$: Node embedding matrix w.r.t. relation r

- Obtain relation specific graph-level representation ($s^{(r)}$: summary vector w.r.t relation r)

$$\mathcal{D}\left(\mathbf{h}_i^{(r)}, \mathbf{s}^{(r)}\right) = \sigma(\mathbf{h}_i^{(r)T} \mathbf{M}^{(r)} \mathbf{s}^{(r)})$$

$$\mathbf{s}^{(r)} = \text{Readout}(\mathbf{H}^{(r)}) = \sigma\left(\frac{1}{n}\sum_{i=1}^{n} \mathbf{h}_i^{(r)}\right)$$

Relation type specific objective

$$\mathcal{L}^{(r)} = \sum_{v_i \in \mathcal{V}}^{n} \log \mathcal{D}\left(\mathbf{h}_i^{(r)}, \mathbf{s}^{(r)}\right) + \sum_{j=1}^{n} \log\left(1 - \mathcal{D}\left(\tilde{\mathbf{h}}_j^{(r)}, \mathbf{s}^{(r)}\right)\right)$$

$$\Rightarrow \quad \mathcal{L} = \sum_{r \in \mathcal{R}} \mathcal{L}^{(r)}$$

However, this cannot capture the interactions among different relation types

Learning Consensus Node Embedding

- How to combine the relation specific embeddings into a single consensus embedding by considering the interactions among different relation types?
  1. **Consensus embedding regularizer**
  2. **Universal discriminator**

Consensus Embedding Regularizer

- Step 1: Aggregate node embeddings from multiple relation types

$$\mathcal{Q}\left(\{\mathbf{H}^{(r)} \mid r \in \mathcal{R}\}\right)$$

- How can we design the aggregation function Q?
  1. Simple summation + average → **Treats all the relation types equivalently**

$$\mathcal{Q}\left(\{\mathbf{H}^{(r)} \mid r \in \mathcal{R}\}\right) = \mathbf{H} = \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} \mathbf{H}^{(r)}$$

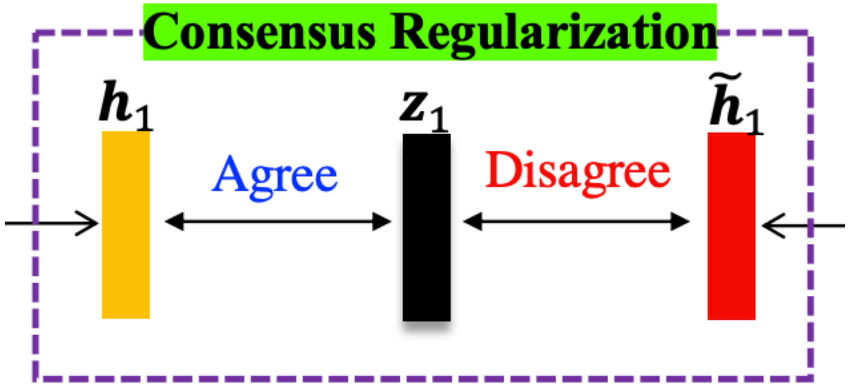
  2. Adopt attention mechanism → **Consider the importance of different relation types**

$$\mathcal{Q}\left(\{\mathbf{h}^{(r)} \mid r \in \mathcal{R}\}\right) = \mathbf{h}_i = \sum_{r \in \mathcal{R}} a_i^{(r)} \mathbf{h}^{(r)}$$

$$a_i^{(r)} = \frac{\exp\left(\mathbf{q}^{(r)} \cdot \mathbf{h}_i^{\text{Concat}}\right)}{\sum_{r' \in \mathcal{R}} \exp\left(\mathbf{q}^{(r')} \cdot \mathbf{h}_i^{\text{Concat}}\right)}$$

Consensus Embedding Regularizer

- Step 2: Introduce a consensus node embedding matrix $\boldsymbol{Z} \in \boldsymbol{R}^{n \times d}$
  - $\boldsymbol{Z}$ should unify all the relation-specific node embeddings
    1) Maximize the agreement with the set of "real" node embeddings
    2) Maximize the disagreement with the "fake" node embeddings



$$\ell_{cs} = \left[\mathbf{Z} - \mathcal{Q}\left(\{\mathbf{H}^{(r)} \mid r \in \mathcal{R}\}\right)\right]^2 - \left[\mathbf{Z} - \mathcal{Q}\left(\{\tilde{\mathbf{H}}^{(r)} \mid r \in \mathcal{R}\}\right)\right]^2$$

Universal Discriminator

- Recall the discriminator $D(\boldsymbol{h}, \boldsymbol{s})$ that discriminates …
  - whether $\boldsymbol{h}$ is from the original graph that can be summarized as $\boldsymbol{s}$

$$\mathcal{D}\left(\mathbf{h}_i^{(r)}, \mathbf{s}^{(r)}\right) = \sigma(\mathbf{h}_i^{(r)T} \boxed{\mathbf{M}^{(r)}} \mathbf{s}^{(r)})$$

Score matrix w.r.t. relation r

$$\mathcal{L}^{(r)} = \sum_{v_i \in \mathcal{V}}^{n} \log \boxed{\mathcal{D}\left(\mathbf{h}_i^{(r)}, \mathbf{s}^{(r)}\right)} + \sum_{j=1}^{n} \log\left(1 - \boxed{\mathcal{D}\left(\tilde{\mathbf{h}}_j^{(r)}, \mathbf{s}^{(r)}\right)}\right)$$

Probability that h_i is from the real graph

Probability that $\tilde{h}_j$ is from the fake graph

- Learn a universal discriminator that is capable of scoring the real pairs higher than the fake pairs **regardless of the relation types**

$$\mathbf{M} = \mathbf{M}^{(1)} = \mathbf{M}^{(2)} = \ldots = \mathbf{M}^{(|\mathcal{R}|)}$$

Facilitates the joint modeling of different relation types together with the consensus regularization

Final Objective

$$\mathcal{L} = \sum_{r \in \mathcal{R}} \mathcal{L}^{(r)} + \alpha \ell_{\text{cs}} + \beta ||\Theta||^2$$

- $L^{(r)}$: Relation-specific loss
- l_{cs}: Consensus regularization framework
- α: Regularization coefficient

$$\mathcal{L}^{(r)} = \sum_{v_i \in \mathcal{V}}^{n} \log \mathcal{D}\left(\mathbf{h}_i^{(r)}, \mathbf{s}^{(r)}\right) + \sum_{j=1}^{n} \log\left(1 - \mathcal{D}\left(\tilde{\mathbf{h}}_j^{(r)}, \mathbf{s}^{(r)}\right)\right)$$

$$\ell_{\text{cs}} = \left[\mathbf{Z} - \mathcal{Q}\left(\{\mathbf{H}^{(r)} \mid r \in \mathcal{R}\}\right)\right]^2 - \left[\mathbf{Z} - \mathcal{Q}\left(\{\tilde{\mathbf{H}}^{(r)} \mid r \in \mathcal{R}\}\right)\right]^2$$

Extension to Semi-supervised Model

- DMGI is trained in a **fully unsupervised manner**
- However, in reality, nodes are sometimes associated with label information, which can guide the training of node embeddings
- **Easily extendable to semi-supervised model**

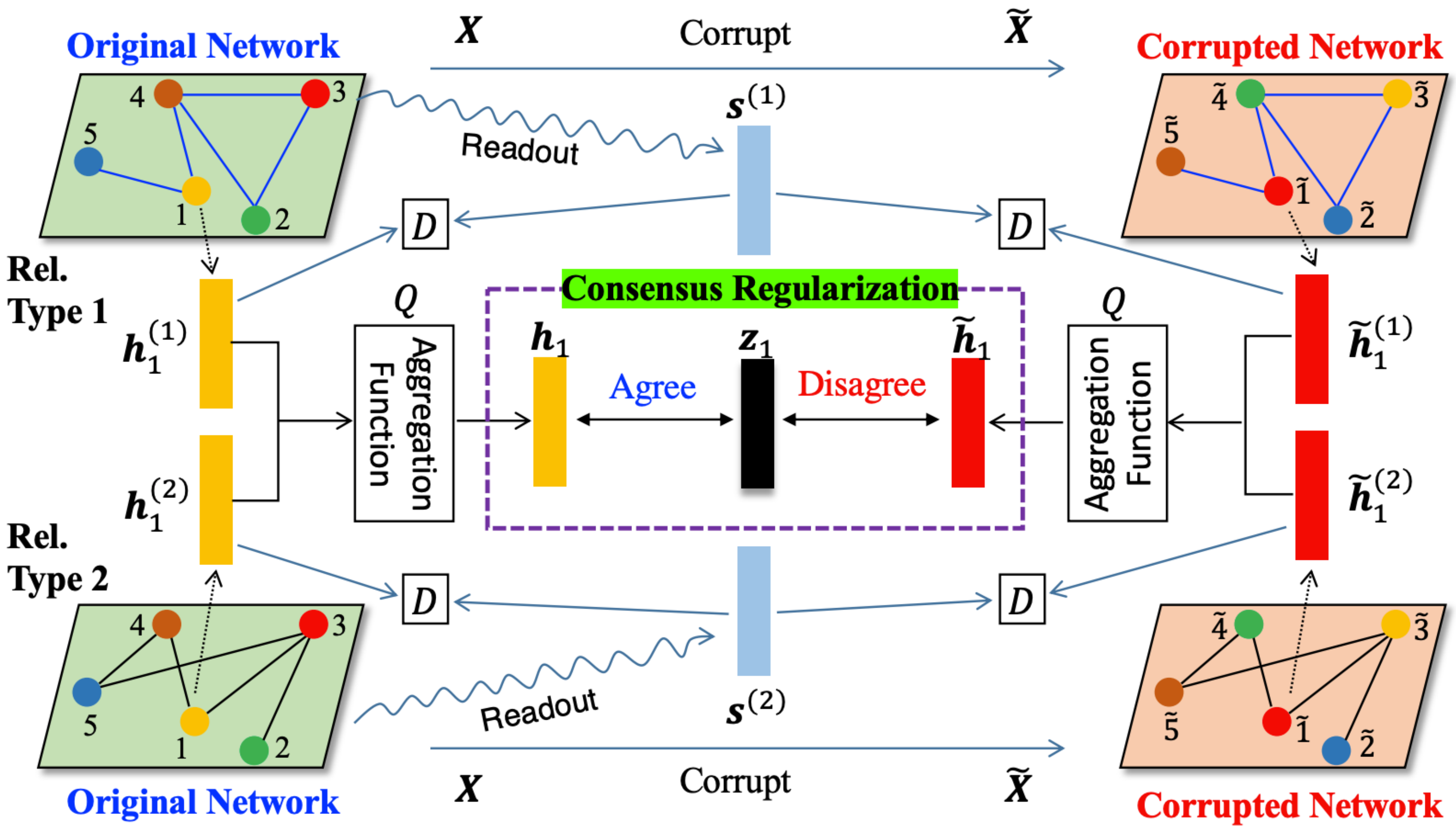
$$\ell_{\text{sup}} = -\frac{1}{|\mathcal{Y}_L|} \sum_{l \in \mathcal{Y}_L} \sum_{i=1}^{c} Y_{li} \ln \hat{Y}_{li}$$

$$\hat{Y} = \text{softmax}(f(\mathbf{Z}))$$

Z: Consensus embedding
$f()$: Logistic regression classifier

$$\boxed{\mathcal{L} = \sum_{r \in \mathcal{R}} \mathcal{L}^{(r)} + \alpha \ell_{\text{cs}} + \beta ||\Theta|| + \gamma \ell_{\text{sup}}}$$

Proposed Framework
: Deep Multiplex Graph Infomax (DMGI)



Experiments

- Dataset

Table 1: Statistics of the datasets. The node attributes are bag-of-words of text associated with each node.

	Relations (A-B)	Num. A	Num. B	Num. A-B	Relation type	Num. relations	Num. node attributes	Num. labeled data	Num. classes
ACM	Paper-Author Paper-Subject	3,025 3,025	5,835 56	9,744 3,025	P-A-P P-S-P	29,281 2,210,761	1,830 (Paper abstract)	600	3
IMDB	Movie-Actor Movie-Director	3,550 3,550	4,441 1,726	10,650 3,550	M-A-M M-D-M	66,428 13,788	1,007 (Movie plot)	300	3
DBLP	Paper-Author Paper-Paper Author-Term	7,907 7,907 1,960	1,960 7,907 1,975	14,238 10,522 57,269	P-A-P P-P-P P-A-T-A-P	144,783 90,145 57,137,515	2,000 (Paper abstract)	80	4
Amazon	Item-Item	7,621	7,621	38,514 45,446 9,783	Also-view Also-bought Bought-together	266,237 1,104,257 16,305	2,000 (Item description)	80	4

Competitors

Table 2: Properties of the compared methods (*Mult.*: Mutliplexity, *Attr*: Attribute, *Unsup*: Unsupervised, *Glo*: Global).

	Mult.	*Attr.*	*Unsup.*	*Glo.*
Dw/n2v	✗	✗	✓	✗
GCN/GAT	✗	✓	✗	✗
DGI	✗	✓	✓	✓
ANRL	✗	✓	✓	✓
CAN	✗	✓	✓	✗
DGCN	✗	✓	✗	✓
CMNA	✓	✗	✓	✓
MNE	✓	✗	✓	✗
mGCN	✓	✓	✓	✗
HAN	✓	✓	✗	✗
DMGI	✓	✓	✓	✓

Evaluation Results

Clustering & Similarity Search (Unsupervised task)

Method	ACM		IMDB		DBLP		Amazon	
	NMI	Sim@5	NMI	Sim@5	NMI	Sim@5	NMI	Sim@5
Deepwalk	0.310	0.710	0.117	0.490	0.348	0.629	0.083	0.726
node2vec	0.309	0.710	0.123	0.487	0.382	0.629	0.074	0.738
GCN/GAT	0.671	0.867	0.176	0.565	0.465	0.724	0.287	0.624
DGI	0.640	0.889	0.182	0.578	0.551	0.786	0.007	0.558
ANRL	0.515	0.814	0.163	0.527	0.332	0.720	0.166	0.763
CAN	0.504	0.836	0.074	0.544	0.323	0.792	0.001	0.537
DGCN	0.691	0.690	0.143	0.179	0.462	0.491	0.143	0.194
CMNA	0.498	0.363	0.152	0.069	0.420	0.511	0.070	0.435
MNE	0.545	0.791	0.013	0.482	0.136	0.711	0.001	0.395
mGCN	0.668	0.873	0.183	0.550	0.468	0.726	0.301	0.630
HAN	0.658	0.872	0.164	0.561	0.472	0.779	0.029	0.495
DMGI	0.687	0.898	**0.196**	**0.605**	0.409	0.766	**0.425**	0.816
$\text{DMGI}_{\text{attn}}$	**0.702**	**0.901**	0.185	0.586	**0.554**	**0.798**	0.412	**0.825**

Node classification (Supervised task)

	ACM		IMDB		DBLP		Amazon	
	MaF1	MiF1	MaF1	MiF1	MaF1	MiF1	MaF1	MiF1
Deepwalk	0.739	0.748	0.532	0.550	0.533	0.537	0.663	0.671
node2vec	0.741	0.749	0.533	0.550	0.543	0.547	0.662	0.669
GCN/GAT	0.869	0.870	0.603	0.611	0.734	0.717	0.646	0.649
DGI	0.881	0.881	0.598	0.606	0.723	0.720	0.403	0.418
ANRL	0.819	0.820	0.573	0.576	0.770	0.699	0.692	0.690
CAN	0.590	0.636	0.577	0.588	0.702	0.694	0.498	0.499
DGCN	0.888	0.888	0.582	0.592	0.707	0.698	0.478	0.509
CMNA	0.782	0.788	0.549	0.566	0.566	0.561	0.657	0.665
MNE	0.792	0.797	0.552	0.574	0.566	0.562	0.556	0.567
mGCN	0.858	0.860	0.623	0.630	0.725	0.713	0.660	0.661
HAN	0.878	0.879	0.599	0.607	0.716	0.708	0.501	0.509
DMGI	**0.898**	**0.898**	**0.648**	**0.648**	0.771	0.766	0.746	0.748
$\text{DMGI}_{\text{attn}}$	0.887	0.887	0.602	0.606	**0.778**	**0.770**	**0.758**	**0.758**

DMGI outperforms all the state-of-the-art baselines not only on the unsupervised tasks, but also the supervised task

(although the improvement is more significant in the unsupervised task as expected)

Evaluation Results

Clustering & Similarity Search (Unsupervised task)

Method	ACM		IMDB		DBLP		Amazon	
	NMI	Sim@5	NMI	Sim@5	NMI	Sim@5	NMI	Sim@5
Deepwalk	0.310	0.710	0.117	0.490	0.348	0.629	0.083	0.726
node2vec	0.309	0.710	0.123	0.487	0.382	0.629	0.074	0.738
GCN/GAT	0.671	0.867	0.176	0.565	0.465	0.724	0.287	0.624
DGI	0.640	0.889	0.182	0.578	0.551	0.786	0.007	0.558
ANRL	0.515	0.814	0.163	0.527	0.332	0.720	0.166	0.763
CAN	0.504	0.836	0.074	0.544	0.323	0.792	0.001	0.537
DGCN	0.691	0.690	0.143	0.179	0.462	0.491	0.143	0.194
CMNA	0.498	0.363	0.152	0.069	0.420	0.511	0.070	0.435
MNE	0.545	0.791	0.013	0.482	0.136	0.711	0.001	0.395
mGCN	0.668	0.873	0.183	0.550	0.468	0.726	0.301	0.630
HAN	0.658	0.872	0.164	0.561	0.472	0.779	0.029	0.495
DMGI	0.687	0.898	**0.196**	**0.605**	0.409	0.766	**0.425**	0.816
$\text{DMGI}_{\text{attn}}$	**0.702**	**0.901**	0.185	0.586	**0.554**	**0.798**	0.412	**0.825**

Node classification (supervised task)

	ACM		IMDB		DBLP		Amazon	
	MaF1	MiF1	MaF1	MiF1	MaF1	MiF1	MaF1	MiF1
Deepwalk	0.739	0.748	0.532	0.550	0.533	0.537	0.663	0.671
node2vec	0.741	0.749	0.533	0.550	0.543	0.547	0.662	0.669
GCN/GAT	0.869	0.870	0.603	0.611	0.734	0.717	0.646	0.649
DGI	0.881	0.881	0.598	0.606	0.723	0.720	0.403	0.418
ANRL	0.819	0.820	0.573	0.576	0.770	0.699	0.692	0.690
CAN	0.590	0.636	0.577	0.588	0.702	0.694	0.498	0.499
DGCN	0.888	0.888	0.582	0.592	0.707	0.698	0.478	0.509
CMNA	0.782	0.788	0.549	0.566	0.566	0.561	0.657	0.665
MNE	0.792	0.797	0.552	0.574	0.566	0.562	0.556	0.567
mGCN	0.858	0.860	0.623	0.630	0.725	0.713	0.660	0.661
HAN	0.878	0.879	0.599	0.607	0.716	0.708	0.501	0.509
DMGI	**0.898**	**0.898**	**0.648**	**0.648**	0.771	0.766	0.746	0.748
$\text{DMGI}_{\text{attn}}$	0.887	0.887	0.602	0.606	**0.778**	**0.770**	**0.758**	**0.758**

DGI shows relatively good performance, but the performance is unstable

(poor performance on Amazon dataset)

→ multiple relation types should be jointly modeled

Evaluation Results

Clustering & Similarity Search (Unsupervised task)

Method	ACM		IMDB		DBLP		Amazon	
	NMI	Sim@5	NMI	Sim@5	NMI	Sim@5	NMI	Sim@5
Deepwalk	0.310	0.710	0.117	0.490	0.348	0.629	0.083	0.726
node2vec	0.309	0.710	0.123	0.487	0.382	0.629	0.074	0.738
GCN/GAT	0.671	0.867	0.176	0.565	0.465	0.724	0.287	0.624
DGI	0.640	0.889	0.182	0.578	0.551	0.786	0.007	0.558
ANRL	0.515	0.814	0.163	0.527	0.332	0.720	0.166	0.763
CAN	0.504	0.836	0.074	0.544	0.323	0.792	0.001	0.537
DGCN	0.691	0.690	0.143	0.179	0.462	0.491	0.143	0.194
CMNA	0.498	0.363	0.152	0.069	0.420	0.511	0.070	0.435
MNE	0.545	0.791	0.013	0.482	0.136	0.711	0.001	0.395
mGCN	0.668	0.873	0.183	0.550	0.468	0.726	0.301	0.630
HAN	0.658	0.872	0.164	0.561	0.472	0.779	0.029	0.495
DMGI	0.687	0.898	**0.196**	**0.605**	0.409	0.766	**0.425**	0.816
$\text{DMGI}_{\text{attn}}$	**0.702**	**0.901**	0.185	0.586	**0.554**	**0.798**	0.412	**0.825**

Node classification (supervised task)

	ACM		IMDB		DBLP		Amazon	
	MaF1	MiF1	MaF1	MiF1	MaF1	MiF1	MaF1	MiF1
Deepwalk	0.739	0.748	0.532	0.550	0.533	0.537	0.663	0.671
node2vec	0.741	0.749	0.533	0.550	0.543	0.547	0.662	0.669
GCN/GAT	0.869	0.870	0.603	0.611	0.734	0.717	0.646	0.649
DGI	0.881	0.881	0.598	0.606	0.723	0.720	0.403	0.418
ANRL	0.819	0.820	0.573	0.576	0.770	0.699	0.692	0.690
CAN	0.590	0.636	0.577	0.588	0.702	0.694	0.498	0.499
DGCN	0.888	0.888	0.582	0.592	0.707	0.698	0.478	0.509
CMNA	0.782	0.788	0.549	0.566	0.566	0.561	0.657	0.665
MNE	0.792	0.797	0.552	0.574	0.566	0.562	0.556	0.567
mGCN	0.858	0.860	0.623	0.630	0.725	0.713	0.660	0.661
HAN	0.878	0.879	0.599	0.607	0.716	0.708	0.501	0.509
DMGI	**0.898**	**0.898**	**0.648**	**0.648**	0.771	0.766	0.746	0.748
$\text{DMGI}_{\text{attn}}$	0.887	0.887	0.602	0.606	**0.778**	**0.770**	**0.758**	**0.758**

Attribute-aware multiplex network embedding methods (mGCN, HAN)
> methods that neglect the node attributes. (CMNA, MNE)

(even though we concatenated node attributes to the node embeddings)

→ verifies not only the benefit of modeling the node attributes, but also that the attributes should be systematically incorporated into the model

Effect of Attention Mechanism

Table 5: Performance of similarity search (Sim@5) of embedding methods for a single network. (Merged denotes the average of all the relation-type specific embeddings.)

ACM		GCN	DGI	ANRL	DMGI	$DMGI_{attn}$
Rel. Type	PAP	0.822	0.875	0.795		
	PSP	0.721	0.675	0.694		
Merged		0.867	0.889	0.814	0.898	**0.901**

IMDB		GCN	DGI	ANRL	DMGI	$DMGI_{attn}$
Rel. Type	MAM	0.485	0.484	0.495		
	MDM	0.548	0.562	0.520		
Merged		0.566	0.578	0.527	**0.605**	0.586

DBLP		GCN	DGI	ANRL	DMGI	$DMGI_{attn}$
Rel. Type	PAP	0.730	0.779	0.692		
	PPP	0.456	0.477	0.680		
	PATAP	0.431	0.409	OOM		
Merged		0.724	0.786	0.720	0.766	**0.799**

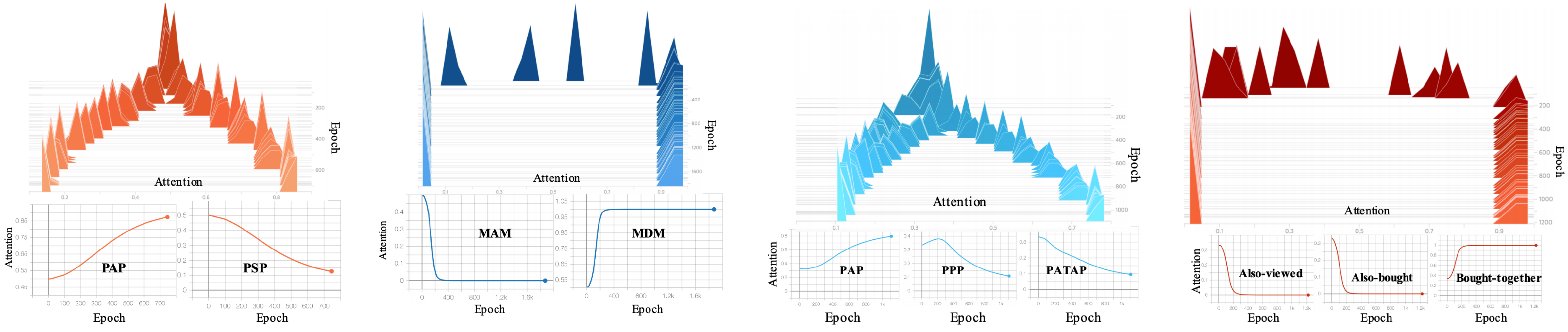
Amazon		GCN	DGI	ANRL	DMGI	$DMGI_{attn}$
Rel. Type	Also-V	0.355	0.367	0.563		
	Also-B	0.357	0.381	0.516		
	Bou.-T	0.662	0.639	0.770		
Merged		0.624	0.558	0.764	0.816	**0.825**

$DMGI_{attn}$ outperforms $DMGI$ in most of the datasets but IMDB dataset

Why?

Analysis on Attention Weights

- Visualization of the attention weights



ACM IMDB DBLP Amazon

- **The attention weights eventually end up in both extremes (Close to either 0 or 1)**
- **Most of the attention weight is dedicated to a single relation type**

Going back …

ACM, DBLP, Amazon

The performance differences among relation types are more biased to a single relation type

→ **Some relations are more important than others**

→ **Skewed attention works**

Table 5: Performance of similarity search (Sim@5) of embedding methods for a single network. (Merged denotes the average of all the relation-type specific embeddings.)

ACM		GCN	DGI	ANRL	DMGI	$\text{DMGI}_{\text{attn}}$
Rel. Type	PAP	0.822	0.875	0.795		
	PSP	0.721	0.675	0.694		
Merged		0.867	0.889	0.814	0.898	**0.901**
IMDB		GCN	DGI	ANRL	DMGI	$\text{DMGI}_{\text{attn}}$
Rel. Type	MAM	0.485	0.484	0.495		
	MDM	0.548	0.562	0.520		
Merged		0.566	0.578	0.527	**0.605**	0.586
DBLP		GCN	DGI	ANRL	DMGI	$\text{DMGI}_{\text{attn}}$
Rel. Type	PAP	0.730	0.779	0.692		
	PPP	0.456	0.477	0.680		
	PATAP	0.431	0.409	OOM		
Merged		0.724	0.786	0.720	0.766	**0.799**
Amazon		GCN	DGI	ANRL	DMGI	$\text{DMGI}_{\text{attn}}$
Rel. Type	Also-V	0.355	0.367	0.563		
	Also-B	0.357	0.381	0.516		
	Bou.-T	0.662	0.639	0.770		
Merged		0.624	0.558	0.764	0.816	**0.825**

IMDB

All the relations show relatively similar performance

→ **Both relations are important**

→ **Skewed attention is not helpful**

Using attention score as a way of filtering

- In DBLP, PATAP turned out to be the most useless relation (i.e., PATAP is noise)

→ We expect that removing PATAP will improve performance

Table 6: NMI on various combinations of relation types.

	DBLP dataset	GCN/GAT	DGI	$\text{DMGI}_{\text{attn}}$
NMI	PAP+PPP	0.464	0.543	**0.565**
	PAP+PATAP	0.458	0.535	0.017
	PPP+PATAP	0.332	0.237	0.201
	All	**0.465**	**0.551**	0.554

- $DMGI_{attn}$ obtains even better results without "PATPA" than using all the relation types, whereas for GCN and DGI, still considering all the relation types shows the best performance.

Attention mechanism can be useful to filter out unnecessary relation types, which will especially come in handy when the number of relation types is large.

Ablation Study

DBLP dataset		MaF1 (Node Classification)	NMI (Clustering)	Sim@5 (Sim. search)
$\text{DMGI}_{\text{attn}}$		0.778	0.554	0.798
1) $\text{DMGI}_{\text{attn}}$+ Semi supervised		0.791	0.555	0.798
2) Readout (Eqn. 3)	Random sample	0.774	0.555	0.797
	Maxpool	0.778	0.552	0.802
	Linear projection	0.783	0.565	0.803
	SAGPool	0.797	0.563	0.797
3) Without 2nd term of Eqn. 6		0.749	0.448	0.787
4) $\mathbf{M} \neq \mathbf{M}^{(1)} \neq \ldots \neq \mathbf{M}^{(\lvert\mathcal{R}\rvert)}$.		0.645	0.076	0.677
5) No attributes (Adj. as attribute)		0.377	0.053	0.763
6) Neg sample: Shuffle adj.		0.364	0.156	0.504

Semi supervised module is mainly beneficial for supervised task

Ablation Study

Node Classification ↓ | Clustering ↓ | Sim. search ↓

DBLP dataset		MaF1	NMI	Sim@5
DMGI$_{\text{attn}}$		0.778	0.554	0.798
1) DMGI$_{\text{attn}}$+ Semi supervised		0.791	0.555	0.798
2) Readout (Eqn. 3)	Random sample	0.774	0.555	0.797
	Maxpool	0.778	0.552	0.802
	Linear projection	0.783	0.565	0.803
	SAGPool	0.797	0.563	0.797
3) Without 2nd term of Eqn. 6		0.749	0.448	0.787
4) $\mathbf{M} \neq \mathbf{M}^{(1)} \neq \ldots \neq \mathbf{M}^{(\lvert\mathcal{R}\rvert)}$.		0.645	0.076	0.677
5) No attributes (Adj. as attribute)		0.377	0.053	0.763
6) Neg sample: Shuffle adj.		0.364	0.156	0.504

Lee et al, 2019 → SAGPool

$$\mathbf{s}^{(r)} = \text{Readout}(\mathbf{H}^{(r)}) = \sigma\left(\frac{1}{n}\sum_{i=1}^{n}\mathbf{h}_i^{(r)}\right)$$

Advanced pooling technique helps, but not significantly

Ablation Study

Node Classification | Clustering | Sim. search

DBLP dataset		MaF1	NMI	Sim@5
$\text{DMGI}_{\text{attn}}$		0.778	0.554	0.798
1) $\text{DMGI}_{\text{attn}}$+ Semi supervised		0.791	0.555	0.798
2) Readout (Eqn. 3)	Random sample	0.774	0.555	0.797
	Maxpool	0.778	0.552	0.802
	Linear projection	0.783	0.565	0.803
	SAGPool	0.797	0.563	0.797
3) Without 2nd term of Eqn. 6		0.749	0.448	0.787
4) $\mathbf{M} \neq \mathbf{M}^{(1)} \neq \dots \neq \mathbf{M}^{(\lvert\mathcal{R}\rvert)}$.		0.645	0.076	0.677
5) No attributes (Adj. as attribute)		0.377	0.053	0.763
6) Neg sample: Shuffle adj.		0.364	0.156	0.504

$$\ell_{\text{cs}} = \left[\mathbf{Z} - \mathcal{Q}\left(\{\mathbf{H}^{(r)} \mid r \in \mathcal{R}\}\right)\right]^2 - \boxed{\left[\mathbf{Z} - \mathcal{Q}\left(\{\tilde{\mathbf{H}}^{(r)} \mid r \in \mathcal{R}\}\right)\right]^2}$$

Maximize the agreement with the set of “real” node embeddings

Maximize the disagreement with the “fake” node embeddings

Ablation Study

Node Classification → MaF1; Clustering → NMI; Sim. search → Sim@5

DBLP dataset		MaF1	NMI	Sim@5
$\mathrm{DMGI}_{\mathrm{attn}}$		0.778	0.554	0.798
1) $\mathrm{DMGI}_{\mathrm{attn}}$+ Semi supervised		0.791	0.555	0.798
2) Readout (Eqn. 3)	Random sample	0.774	0.555	0.797
	Maxpool	0.778	0.552	0.802
	Linear projection	0.783	0.565	0.803
	SAGPool	0.797	0.563	0.797
3) Without 2nd term of Eqn. 6		0.749	0.448	0.787
4) $\mathbf{M} \neq \mathbf{M}^{(1)} \neq \ldots \neq \mathbf{M}^{(\lvert\mathcal{R}\rvert)}$.		0.645	0.076	0.677
5) No attributes (Adj. as attribute)		0.377	0.053	0.763
6) Neg sample: Shuffle adj.		0.364	0.156	0.504

Universal discriminator is critical

Conclusion

- A simple yet effective unsupervised method for embedding attributed multiplex network
- DMGI can jointly integrate the embeddings from multiple types of relations between no des through the **consensus regularization framework**, and the **universal discriminator**
- The attention mechanism can infer the importance of each relation type
  - Facilitates the preprocessing of the multiplex network
- Showed superior results not only on unsupervised tasks, but also on a supervised task

References

- [Qu et al, 2017] Qu, M.; Tang, J.; Shang, J.; Ren, X.; Zhang, M.; and Han, J. 2017. An attention-based collab oration framework for multi-view network representation learning. In CIKM. ACM
- [Zhang et al, 2018] Zhang, Z.; Yang, H.; Bu, J.; Zhou, S.; Yu, P.; Zhang, J.; Ester, M.; and Wang, C. 2018b. Anr l: Attributed network representation learning via deep neural networks. In IJCAI
- [Chu et al, 2019] Chu, X.; Fan, X.; Yao, D.; Zhu, Z.; Huang, J.; and Bi, J. 2019. Cross-network embedding for multi-network alignment. In WWW
- [Hjelm et al, 2019] Hjelm, R. D.; Fedorov, A.; Lavoie-Marchildon, S.; Grewal, K.; Trischler, A.; and Bengio, Y. 2019. Learning deep representations by mutual information estimation and maximization. ICLR
- [Velickovic et al, 2019] Velickovic, P.; Fedus, W.; Hamilton, W. L.; Li ´ o, P.; Bengio, ` Y.; and Hjelm, R. D. 201 9. Deep graph infomax. ICLR
- [Ma et al, 2019] Ma, Y.; Wang, S.; Aggarwal, C. C.; Yin, D.; and Tang, J. 2019. Multi-dimensional graph conv olutional networks. In SDM. SIAM
- [Wang et al, 2019] Wang, X.; Ji, H.; Shi, C.; Wang, B.; Ye, Y.; Cui, P.; and Yu, P. S. 2019. Heterogeneous graph attention network. In WWW. ACM.
- [Lee et al, 2019] Lee, J.; Lee, I.; and Kang, J. 2019. Self-attention graph pooling. ICML.