Modeling Heterogeneous Edges to Represent Networks with Graph Auto-Encoder

Lu Wang\textsuperscript{*1,2,3}, Yu Song\textsuperscript{*1,2,3}, Hong Huang\textsuperscript{**1,2,3}, Fanghua Ye\textsuperscript{4}, Xuanhua Shi\textsuperscript{1,2,3}, and Hai Jin\textsuperscript{1,2,3}

\textsuperscript{1} National Engineering Research Center for Big Data Technology and System
\textsuperscript{2} Service Computing Technology and Systems Laboratory
\textsuperscript{3} School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China
\textsuperscript{4} School of Computer Science, University College London, London, United Kingdom

wluluo@gmail.com, yusonghust@hust.edu.cn, honghuang@hust.edu.cn, smartyfh@outlook.com, xhshi@hust.edu.cn, hjin@hust.edu.cn

Abstract. In the real world, networks often contain multiple relationships among nodes, manifested as the heterogeneity of the edges in the networks. We convert the heterogeneous networks into multiple views by using each view to describe a specific type of relationship between nodes, so that we can leverage the collaboration of multiple views to learn the representation of networks with heterogeneous edges. Given this, we propose a regularized graph auto-encoders (RGAE) model, committed to utilizing abundant information in multiple views to learn robust network representations. More specifically, RGAE designs shared and private graph auto-encoders as main components to capture high-order nonlinear structure information of the networks. Besides, two loss functions serve as regularization to extract consistent and unique information, respectively. Concrete experimental results on realistic datasets indicate that our model outperforms state-of-the-art baselines in practical applications.

Keywords: Network Embedding · Network Analysis · Deep Learning

1 Introduction

The research of network analysis has made rapid progress in recent years. In fact, network data are usually complex and therefore hard to process. To mine network data, one fundamental task is to learn a low-dimensional representation for each node, such that network properties are preserved in the vector space. As a result, various downstream applications, such as link prediction \cite{22}, classification \cite{36}, and community detection\cite{10}, can be directly conducted in such vector space. As for learning representations for networks, there are two main challenges that have not yet been fully resolved:

(1) **Preservation of heterogeneous relationships between nodes.** There usually exist diverse and different types of relationships between nodes, leading to the heterogeneity
of edges. For example, in the twitter network, four types of relationships may be observed in the interactions between two users, that is one user may retweet, reply, like, and mention another user’s tweet. Thus it is reasonable to build four types of edges between the two users with each type of edge corresponding to one type of relationship. Although these edges reflect the similarity between the two users, we can not ignore the slight difference at the “semantic” level. Therefore, taking heterogeneity of edges into consideration for representing such networks is quite significant. In literature, several heterogeneous network embedding approaches (e.g. PTE [29], Metapath2vec [4], and HIN2Vec[7]) have been proposed to represent heterogeneous nodes or edges into the same semantic vector space. However, these methods only learn a final representation for all relationships jointly but ignore the different semantic meanings of edges. Therefore, in order to explore the heterogeneity of edges, it is necessary to learn a relation-specific representation for each type of relationship.

(2) Preservation of high-order node proximities. As described in LINE [30], it defines two loss functions to preserve both 1-st and 2-nd order proximities together. However, it is also meaningful to further integrate the information of k-th-order neighbors for enhancing the representation of nodes with small degrees. Moreover, most existing network embedding methods are equivalent to implicit matrix factorization [23], which is a shallow model that fails to capture high-order nonlinear proximities between nodes. GraRep [2] aims to capture the k-th-order proximity by factorizing the k-step (k=1,2,···,K) transition matrices. However, the matrix factorization technique is usually time inefficient and hard to learn nonlinear relationships between nodes. SDNE [34] designs a deep auto-encoder framework to extract the nonlinear structural information of networks, but it still only considers 1-st and 2-nd order proximities without preserving even higher order proximities between nodes. Consequently, to preserve the complex network information, a better solution should leverage high-order nonlinear structural information to yield more robust network representations.

Recently, it has witnessed that multi-view learning is applied successfully in a wide variety of applications, especially for mining heterogeneous data, such as clustering [13], computer vision [16], and information retrieval [21]. In this regard, we convert heterogeneous edges into multiple views for a network, and solving a multi-view learning problem to learn representations for such networks. To be more specific, we abstract each relationship as a view of the network, reflecting a type of proximity between nodes, thus the original network can be further interpreted as a multi-view network. Finally, we formalize the task as a multi-view network embedding problem. Existing multi-view network embedding methods, such as MVE [24] and MINEs [19], first learn a single-view network representation using skip-gram model then fuse them directly. Since their fusion strategies, i.e. averaging and adding, are both linear functions, they fail to capture the complex nonlinear information, leading to a sub-optimal result. Besides, there are some works [27,37,40] learning a unified representation and a view-specific representation for each view simultaneously, but they are shallow models without considering the high-order proximities between nodes.

Targeting at modeling the heterogeneity of edges and preservation of high-order node proximities for learning network representations, we propose a novel Regularized Graph Auto-Encoders framework, namely RGAE. To better illustrate our motivation,
we first introduce a case study on a multi-view AMiner network (see details in sec 4.1). As shown in Fig. 1(a), it contains two types of information, consistent information and unique information, as its edges are partial aligned as well as partial distinct between different views. Different views may share some consistent information. At the same time, each of them also carries some unique information that others do not have. We further follow a similar method [27] to perform a statistical analysis. Given a pair of views, the edge sets are $E_1$ and $E_2$. We treat the Jaccard coefficient between the two sets as the proportion of consistent information. As we can see in Fig. 1(b), there exists noticeable consistent information between coauthor and text similarity views while other pairs of views are quite negligible. Thus we conclude that it is unreasonable to preserve only consistent or unique information for multi-view network embedding. As a result, RGAE model aims to preserve consistent and unique information simultaneously, as well as capturing high-order nonlinear proximities between nodes. The contributions of our model are threefold:

(1) In consideration of preserving heterogeneous information of edges as much as possible, we design two kinds of graph auto-encoders to deal with consistent and unique information respectively: one is the shared across view and the other is private to each view. Through these deep and nonlinear graph auto-encoders, our RGAE model is able to represent complex high-order structural information.

(2) We further introduce two regularized loss functions, i.e. the similarity loss and the difference loss, to explicitly avoid the information redundancy of the two types of graph auto-encoders. The similarity loss is used to extract consistent information from shared graph auto-encoders. The difference loss aims to encourage the independence between shared and private graph auto-encoders, so the unique information can also be well preserved at the same time.

(3) To evaluate the performance of the RGAE model, we conduct abundant experiments on four real-world datasets. The experimental results demonstrate that the proposed model is superior to existing state-of-the-art baseline approaches as well as examining the novelty of our model.

2 Problem Formulation and Notations

We first briefly define a multi-view network, multi-view network embedding and list the main notations used throughout this paper in Table 1.
Table 1. Summary of symbols

| Symbol | Definition | Symbol | Definition |
|--------|------------|--------|------------|
| $\mathcal{U}$ | node set | $\mathcal{E}_i$ | edge set of view $i$ |
| $|V|$ | number of views | $D$ | dimension of $Y$ |
| $N$ | number of nodes | $d$ | $\left\lfloor \frac{D}{(|V| + 1)} \right\rfloor$ |
| $A_i \in \mathbb{R}^{N \times N}$ | adjacency matrix of view $i$ | $\alpha, \beta, \gamma$ | hyper-parameters |
| $Y_{i,p} \in \mathbb{R}^d$ | private embedding of view $i$ | $I_N \in \mathbb{R}^{N \times N}$ | an identity matrix |
| $Y_{i,s} \in \mathbb{R}^d$ | shared embedding of view $i$ | $\tilde{A}_i = A_i + I_N$ |
| $Y_{con} \in \mathbb{R}^d$ | consistent embedding | $\tilde{D}_i(m,m) = \sum_n A_i(m,n)$ |
| $Y \in \mathbb{R}^D$ | final network embedding | $X_1 \oplus X_2$ | concatenation in the last dimension |

Definition 1. **Multi-View Network** A multi-view network is a network defined as $G = \{\mathcal{U}, \mathcal{E}_1, \mathcal{E}_2, \cdots, \mathcal{E}_{|V|}\}$, where $\mathcal{U}$ is a node set shared by all views, and $\mathcal{E}_i$ ($1 \leq i \leq |V|$) is the edge set of the $i$-th view, which reflects a specific type of relationship between nodes.

Problem 1. **Multi-View Network Embedding** Given a multi-view network $G = \{\mathcal{U}, \mathcal{E}_1, \mathcal{E}_2, \cdots, \mathcal{E}_{|V|}\}$, the multi-view network embedding problem aims to learn a low-dimensional embedding representation $Y \in \mathbb{R}^D$ ($D \ll N$). More specifically, an intermediate view-specific embedding representation $Y_{i,p} \in \mathbb{R}^d$ is learned to preserve the unique information of view $i$ and a shared embedding representation $Y_{con} \in \mathbb{R}^d$ is learned to preserve the consistent information among all views. The final embedding representation $Y$ is obtained from all view-specific embedding representations and the shared embedding representation by an aggregation function.

3 Method

Fig. 2. The framework of RGAE. The illustration takes a network with three views as an example.

In this section, we introduce our proposed **Regularized Graph Auto-Encoders** framework, namely RGAE, for tackling the multi-view network embedding problem in detail. An illustrative example of the RGAE model is shown in Fig. 2.

3.1 The Shared and Private Graph Auto-Encoders

Graph convolutional network (GCN) [11] is built on the idea of message passing, and convolves the representation of the central node with the representations of its neighbors...
to derive an updated representation of the central node. Our shared and private graph auto-encoders are both motivated as an extension of existing GCN that is able to learn valuable information for graphs. By stacking multiple GCN layers as an encoder and a simple inner production operation as a decoder, the graph auto-encoders in the RGAE model is capable of extracting consistent and unique information in a multi-view network. Specifically, given a multi-view network denoted as $G = \{U, E_1, E_2, \cdots, E_{|V|}\}$, for a specific view $i$, the propagation rule of $l$-th layer in the private encoder is formulated as:

$$Y_{i,p}^{(l+1)} = \sigma(\tilde{D}_i^{-\frac{1}{2}} \tilde{A} \tilde{D}_i^{-\frac{1}{2}} Y_{i,p}^{(l)} W^{(l)})$$

(1)

where the $\sigma(\cdot)$ is the non-linear activation function. In this paper, we choose $relu$ as activation function in all cases. $W^{(l)}$ is the weight matrix, and $Y_{i,p}^{(0)} = X_i$ is the feature matrix for view $i$. Specially, if the node features are not available the $X_i$ will be an identity matrix, as described in [11].

The key point of the shared encoder is that the weight matrices in all layers are shared across different views $^5$ which is clearly different from the private graph auto-encoder. In detail, the propagation rule of the $l$-th layer in the shared graph encoder is formulated as:

$$Y_{i,s}^{(l+1)} = \sigma(\tilde{D}_i^{-\frac{1}{2}} \tilde{A} \tilde{D}_i^{-\frac{1}{2}} Y_{i,s}^{(l)} W^{(l)})$$

(2)

Note that the weight matrix $W^{(l)}$ is only shared in view-wise rather than layer-wise. Through this shared architecture we can project all views into the same semantic space so that the process of extracting the consistent information is more interpretable. We can also allow different views to influence mutually and collaborate implicitly.

The GCN layer is motivated by a first-order approximation of the localized spectral filters on graph-structured data [3]. In this regard, it is possible to stack multiple GCN layers in both shared encoders and private encoders to capture the high-order proximity between nodes. The final outputs of these stacked shared encoders and private encoders are denoted as $Y_{i,s}$ and $Y_{i,p}$ for each view respectively. During the forward pass, the graph decoder in view $i$ aims to calculate the reconstructed adjacency matrix $\hat{A}_i$. In order to utilize the complete information to make a better reconstruction, we first concatenate the outputs of the shared encoder and private encoder for view $i$, then we utilize the inner production operation to yield the reconstructed adjacency matrix, as described in [12], which is computed as follow:

$$Y_i = Y_{i,s} \oplus Y_{i,p}, \quad \hat{A}_i = sigmoid(Y_i Y_i^T)$$

(3)

Since the adjacency matrix preserves the topology information of the graph, it is momentous to minimize the reconstruction loss. It has been demonstrated that minimizing the reconstruction loss is helpful to preserve the similarity between nodes [26]. Due to the sparsity of networks, there exist a great deal of zero elements and the number of zero elements and non-zero elements is extremely unbalanced in the adjacency matrix. As a result, we minimize the reconstruction error by optimizing the Balanced Cross-Entropy loss, which allows the model to pay more attention to the non-zero elements thus ignores the redundant noises from zero elements. For the view $i$, we compute

$^5$ Note the node set is shared across all views.
the reconstruction loss as follows:

\[ L_{rec}^{i} = \sum_{a_{i}^{(m,n)} \in A_{i}, \hat{a}_{i}^{(m,n)} \in \hat{A}_{i}} \left[ -a_{i}^{(m,n)} \log(\hat{a}_{i}^{(m,n)}) \varsigma - (1 - a_{i}^{(m,n)}) \log(1 - \hat{a}_{i}^{(m,n)}) \right] \] (4)

where the \( \varsigma \) is a weighting factor to balance the importance of the non-zero elements, defined as \( \frac{\#\text{zero elements}}{\#\text{non-zero elements}} \) in \( A_{i} \).

3.2 Regularization

**Similarity Loss** Intuitively, the consistent information can be extracted from the outputs of the shared encoders. Since we have projected all these outputs into the same semantic space, it is meaningful to make them collaborate to vote for the consistent representation. In this process, we encourage the consistent representation \( Y_{con} \) to be similar to the shared representation \( Y_{i,s} \) of each view as much as possible. As the importance of views may be different, we further allow the model to assign different weights to them. Taking all these into consideration, we introduce the following similarity loss to regularize the extraction process:

\[ L_{sim} = \frac{|V|}{\sum_{i=1}^{\left| V \right|} \lambda_{i}} \| Y_{con} - Y_{i,s} \|_{F}^{2}, \quad \sum_{i=1}^{\left| V \right|} \lambda_{i} = 1, \lambda_{i} \geq 0 \] (5)

where \( \lambda_{i} \) is the weight for view \( i \), and \( \gamma \) moderates the weight distribution. By learning proper weights, the extraction process can let the consistent representation focus on the most informative views. Naturally, the consistent representation is calculated as the weighted combinations of the outputs of the shared encoders, which illustrates the collaboration between different views.

**Difference Loss** In order to preserve the unique information, the difference loss is also introduced to encourage the isolation between consistent embeddings and unique embeddings. As the consistent information and unique information have essential differences, they should be distinguished clearly to avoid the information redundancy. In other words, the shared embeddings and private embeddings should describe the information of multiple views in different perspectives, thus we define the difference loss via an orthogonality constraint between the private embedding and shared embedding in each view:

\[ L_{diff}^{i} = \| Y_{i,s} \odot Y_{i,p} \|_{F}^{2}, \quad i = 1, 2, \ldots, \left| V \right| \] (6)

where the \( \odot \) is the row-wise inner production. Obviously, the difference loss will drive the shared embeddings to be orthogonal with the private embeddings, thus they will be as dissimilar as possible. In this way, the shared and private encoders are able to encode different aspects of the multi-view network. In this paper, we treat the output of the private graph encoder for each view as its private representation.
3.3 The Aggregation Process

As introduced above, our RGAE model includes the three types of losses, i.e. the reconstruction loss, the similarity loss, and the difference loss. In order to train these losses jointly, the overall loss of our proposed model is summarized as follow:

$$L = \sum_{i=1}^{\lvert V \rvert} L^\text{rec}_i + \alpha \cdot L^\text{sim} + \beta \cdot \sum_{i=1}^{\lvert V \rvert} L^\text{dif}_i$$  \hspace{1cm} (7)

where $\alpha$ and $\beta$ are hyper-parameters to control the importance of similarity loss and difference loss respectively. Up to now, we have obtained the representations of consistent and unique information. Finally, we design an aggregation process to yield the final network representation, which can be illustrated as:

$$Y = \text{Aggregator}(Y_{\text{con}}, Y_{1,p}, \cdots, Y_{\lvert V \rvert,p})$$  \hspace{1cm} (8)

The aggregator should be able to integrate both the consistent and unique information effectively, and it can be add, average, pooling and some other designed functions. In this paper, we choose concatenation as the aggregation function since it has been proven to be useful and efficient in many existing network embedding methods [15,27,30]. As shown in Table 1, the total dimension $D$ has been assigned to each graph auto-encoder equally, thus after the concatenation process the final network embedding will still satisfy $Y \in \mathbb{R}^D$.

3.4 Implementation

In practice, we utilize Tensorflow for an efficient GPU-based implementation of the RGAE model. Then the parameters of RGAE model except $\lambda_i$ can be efficiently optimized automatically with back propagation algorithm. To save space, we omit details here. Since the sparsity of network data, we use sparse-dense matrix multiplication for Eqs. (1) and (2), as described in [11]. Specially, for the view weight $\lambda_i$ in Eq. (5), we follow the same method [1] to update it. Let’s denote $\|Y_{\text{con}} - Y_{i,s}\|_F^2$ as $B_i$, then Eq. (5) is equivalent to $\sum_{i=1}^{\lvert V \rvert} \lambda_i B_i - \xi (\sum_{i=1}^{\lvert V \rvert} \lambda_i - 1)$, where $\xi$ is Lagrange multiplier. By taking the derivative of this formula with respect to $\lambda_i$ as zero, we can obtain the update rule of $\lambda_i$: $\lambda_i \leftarrow \frac{(\gamma B_i)^\top}{\sum_j (\gamma B_j)^\top}$. It is efficient to use one parameter $\gamma$ for controlling the distribution of view weights during the optimization process dynamically. According to the update rule, we would assign equal weights to all views when $\gamma$ closes to $\infty$. When $\gamma$ closes to 1, the weight for the view whose $B_i$ value is smallest will be assigned as 1, while others are almost ignored since their weights are close to 0. The pseudo code is shown in Algorithm [1].

4 Experiments

4.1 Experimental Setup

We select four multi-view network datasets in different fields. The statistic analysis is shown in Table 2.
Algorithm 1: The pseudo-code of RGAE Model

```
Input: \( G = \{ \{U, E_1, ..., E_V\}, D, K, \alpha, \beta, \gamma \} \) and hidden size for each encoder layer; 
Output: The network representation \( Y \) for the network \( G \).

\begin{algorithm}
\For {i=1, 2, ..., |V|} {
    According to Eq. (1), construct private graph encoder for view \( i \);
    According to Eq. (2), construct shared graph encoder for view \( i \);
    According to Eqs. (3) and (4), construct graph decoder and calculate \( L_{rec}^i \);
}
According to Eq. (5) and Eq. (6), calculate \( L_{sim}^i \) and \( L_{dif}^i \);
\Repeat {
    Update parameters in RGAE model via optimizing the \( L \) in Eq. (7);
    until Convergence;
}
According to Eq. (8), obtain the final network representation \( Y \);
\Return Network representation \( Y \).
\end{algorithm}
```

Table 2. Overview of datasets

| Task                        | Dataset | Views | Nodes     | Edges          | Labels | Type    |
|-----------------------------|---------|-------|-----------|----------------|--------|---------|
| Multi-class Node Classification | AMiner  | 3     | 8,438     | 2,433,356      | 8      | Academic |
| Multi-class Node Classification | PPI     | 6     | 4,328     | 1,661,756      | 50     | Biological |
| Multi-label Node Classification | Flickr  | 2     | 34,881    | 3,290,030      | 171    | Social   |
| Link Prediction             | YouTube | 4     | 5,108     | 3,263,045      | -      | Social   |

- AMiner [31]: AMiner network is an academic network representing the relationships between authors. It consists of three views: author-citation, co-authorship, and text similarity. Text similarity between two authors is calculated by TF-IDF from titles and abstracts in their papers. An author establishes connections with his top ten similar authors and we only preserve authors in eight research fields as [4]. The research fields are treated as node labels.
- PPI [5]: The PPI network is a human protein-protein interaction network. Six views are constructed based on the co-expression, co-occurrence, database, experiment, fusion, and neighborhood information. Gene groups are treated as node labels.
- Flickr [32]: It is a social network of online users on Flickr with two views. One view is the friendship network among bloggers. The other is a tag-proximity network in which a node connects with its top 10 similar nodes according to their tags. We treat community memberships as node labels.
- YouTube [38]: It is a social network consists of four views: the friendship, the number of common friends, the number of common subscribers, and the number of common favorite videos between two users.

In order to evaluate the effectiveness of RGAE, we compare our model with three types of baselines. The single-view based baselines include:
- Deepwalk [22]: It is a well-known baseline for network embedding. We set the number and the length for each node as 80 and 40 respectively following the recommendations of the original paper. The window-size is set as 10.
- GraRep [2]: It aims to capture the k-order proximities by factorizing the k-step transition matrices. We set k as 5.
- SDNE [34]: It utilizes the auto-encoders to preserve the neighbor structure of nodes. The first-order and second-order proximity are proposed to preserve the global and the local network structure. We set the number of layers as 3, and the hidden size as [800, 400, 128].
- GAE [12]: It stacks GCN layers as an encoder and the inner production operation as a decoder. The reconstruction loss helps it to capture structural information in an unsupervised manner. We set the number of layers and hidden sizes same as SDNE.

The heterogeneous network embedding methods include:
- PTE [29]: It is a heterogeneous network embedding method which can also be used to jointly train the embedding, because multi-view network is a special type of heterogeneous network. We set the number of negative samples as 5.
- Metapath2vec [4]: It utilizes meta-paths guided random walk to generate the node sequences then uses skip-gram model to learn the node representations. We set the number, the length of walks and window size same as deepwalk. We perform experiment using one of all possible meta-paths at a time, and report the best result.

The multi-view based baselines include:
- Deepwalk-con: It applies Deepwalk to get a d dimensional representation for each view then concatenates these representations from all K views to generate a unified representation with K * d dimensions.
- MultiNMF [17]: It is a multi-view matrix factorization algorithm, which extracts consistent information by a joint matrix factorization process.
- MVE [24]: It combines single view embeddings by weights learned from attention mechanism to construct a multi-view network embedding. We set the parameters of random walk and skip-gram model same as Deepwalk, and other parameters are same as the original paper.
- MNE [40]: It combines the information of multiple view by preserving a high dimensional common embedding and a lower dimensional embedding for each view. The dimensions of the additional vectors are set as 10.
- MTNE-C [37]: It combines the common embedding and node-specific embedding of each node to be a complete embedding for the closeness measurement. We follow the default parameter setting in the original paper.

For RGAE and all baselines except Deepwalk-con, the embedding dimension is set as 128. The number of graph auto-encoder layers is set as 3, and two hidden layers’ dimensions are set as 800 and 400 respectively. Both \( \alpha \) and \( \beta \) are selected from \{0.1, 0.3, 0.5, 0.7, 1.0, 1.5, 2.0\}, and \( \gamma \) is selected from \{0.05, 0.5, 5, 10, 50, 100, 500\}. The learning rate is selected from \{0.001, 0.01, 0.1\}. As node features are not available for our datasets the feature matrix will be an identity matrix. We treat the node embedding learned by various methods as feature to train linear classifiers for multiclass classification, and train one-vs-rest classifiers for multilabel classification. For link prediction, we use the cosine similarity between node pairs as features to train a logistic classifier to predict the link existence. Follow the setting in [24], we use other three views to train embeddings and predict the link existence in friend view. To generate negative edges, we randomly sample an equal number of node pairs which have no edge connecting them. We report the best results among multiple views for single-view based baselines.
To guarantee a fair comparison, we repeat each method ten times and the average metrics are reported.

Table 3. Node classification results w.r.t. Micro-F1(%) and Macro-F1(%) with different training ratio. '-' means out of memory error.

| Datasets | Category         | Methods       | 0.1 Micro | 0.1 Macro | 0.3 Micro | 0.3 Macro | 0.5 Micro | 0.5 Macro |
|----------|-----------------|---------------|-----------|-----------|-----------|-----------|-----------|-----------|
| AMiner   | Single-View     | Deepwalk      | 69.9      | 68.4      | 74.3      | 73.3      | 75.1      | 74.3      |
|          |                 | GraRep        | 23.3      | 20.5      | 44.8      | 41.8      | 61.9      | 60.6      |
|          |                 | SDNE          | 64.8      | 62.5      | 70.3      | 68.0      | 70.8      | 69.4      |
|          |                 | GAE           | 60.4      | 54.8      | 62.5      | 57.7      | 63.6      | 59.3      |
|          | Heterogeneous   | PTE           | 52.9      | 46.9      | 56.6      | 52.7      | 58.1      | 55.2      |
|          |                 | Metapath2Vec  | 70.6      | 70.1      | 75.3      | 73.5      | 76.2      | 74.9      |
|          | Multi-View      | Deepwalk-con  | 61.4      | 59.0      | 74.2      | 72.6      | 76.1      | 74.9      |
|          |                 | MultiNMF      | 57.4      | 52.6      | 66.4      | 64.1      | 66.8      | 62.8      |
|          |                 | MVE           | 73.6      | 72.7      | 78.8      | 77.5      | 78.9      | 77.6      |
|          |                 | MNE           | 73.6      | 72.2      | 79.2      | 77.8      | 79.6      | 78.1      |
|          |                 | MTNE-C        | 54.5      | 48.8      | 57.2      | 53.9      | 58.6      | 55.2      |
|          |                 | RGAE          | 74.9      | 73.3      | 80.6      | 79.7      | 82.0      | 80.9      |
| PPI      | Single-View     | Deepwalk      | 8.9       | 4.2       | 10.9      | 6.1       | 12.1      | 7.3       |
|          |                 | GraRep        | 4.0       | 2.0       | 5.1       | 3.1       | 13.1      | 10.0      |
|          |                 | SDNE          | 11.8      | 10.7      | 14.7      | 13.4      | 17.6      | 15.0      |
|          |                 | GAE           | 9.5       | 4.3       | 12.3      | 8.0       | 13.7      | 9.1       |
|          | Heterogeneous   | PTE           | 12.8      | 9.5       | 19.7      | 11.7      | 22.0      | 14.0      |
|          |                 | Metapath2Vec  | 13.4      | 10.0      | 20.2      | 12.8      | 22.3      | 15.7      |
|          | Multi-View      | Deepwalk-con  | 9.9       | 6.2       | 11.9      | 8.5       | 13.6      | 9.9       |
|          |                 | MultiNMF      | 15.3      | 11.9      | 17.8      | 15.2      | 20.3      | 17.5      |
|          |                 | MVE           | 11.7      | 9.9       | 12.1      | 10.6      | 13.3      | 10.8      |
|          |                 | MNE           | 13.3      | 11.8      | 14.1      | 12.2      | 15.6      | 12.1      |
|          |                 | MTNE-C        | 3.4       | 1.6       | 4.0       | 2.0       | 6.2       | 3.5       |
|          |                 | RGAE          | 19.0      | 15.1      | 24.4      | 21.0      | 25.0      | 21.3      |
| Flickr   | Single-View     | Deepwalk      | 51.7      | 32.1      | 51.9      | 27.6      | 53.2      | 27.8      |
|          |                 | GraRep        | 52.4      | 32.2      | 53.8      | 35.0      | 55.9      | 35.8      |
|          |                 | SDNE          | 47.6      | 32.1      | 48.2      | 32.6      | 49.6      | 30.5      |
|          |                 | GAE           | 34.5      | 9.1       | 37.0      | 10.4      | 38.4      | 11.1      |
|          | Heterogeneous   | PTE           | 55.7      | 30.4      | 56.4      | 34.3      | 56.2      | 31.0      |
|          |                 | Metapath2Vec  | 55.7      | 30.8      | 56.6      | 33.9      | 56.7      | 32.2      |
|          | Multi-View      | Deepwalk-con  | 51.9      | 32.6      | 52.5      | 28.2      | 53.7      | 28.3      |
|          |                 | MultiNMF      | -         | -         | -         | -         | -         | -         |
|          |                 | MVE           | 52.0      | 32.5      | 53.0      | 28.9      | 54.3      | 28.8      |
|          |                 | MNE           | 52.4      | 33.1      | 53.5      | 29.9      | 54.8      | 29.8      |
|          |                 | MTNE-C        | 23.9      | 5.2       | 23.3      | 4.8       | 22.9      | 4.6       |
|          |                 | RGAE          | 56.7      | 32.9      | 57.6      | 33.7      | 58.4      | 36.2      |
4.2 Experimental Results

Node Classification We evaluate the performance of our method and three categories of baselines using the Micro-F1 and Macro-F1 scores. Table 3 shows the comparison on three datasets. As can be seen, our RGAE model outperforms all baselines except for Macro-F1 on Flickr dataset. For example, on AMiner dataset, it achieves a sustainable performance gain of 1%, 2%, and 3% with the percentage of training data increasing. It is noted that RGAE always outperforms GAE consistently, which shows that with making good use of information from multiple views, we are indeed able to learn a robust representation for a multi-view network. The superiority of RGAE over SDNE further verifies that it is reasonable to model the heterogeneity of edges. Although GraRep captures high order proximities between nodes, the matrix factorization process makes it hard to preserve non-linear network information, which is not compared with our model. One may see that the existing multi-view network embedding approaches are also not comparable to our RGAE model. The reason is that either they are not possible to consider the uniqueness of each view, like Metapath2Vec and MVE, or they are not possible to capture high-order proximities between nodes, such as MTNE-C and MNE. All these observed results show that the RGAE model can indeed capture more complete non-linear information from multiple views.

Table 4. Link Prediction results on YouTube dataset w.r.t. ROC_AUC Score(%) and Average Precision Score (AP)(%) with different training ratio

| Category       | Methods         | 0.1 ROC_AUC | AP | 0.3 ROC_AUC | AP | 0.5 ROC_AUC | AP |
|----------------|-----------------|-------------|----|-------------|----|-------------|----|
| Single-View    | Deepwalk        | 74.4 73.6   |    | 74.7 74.0   |    | 78.4 77.2   |    |
|                | GraRep          | 80.2 79.6   |    | 80.3 79.8   |    | 80.7 80.0   |    |
|                | SDNE            | 81.8 82.3   |    | 83.0 83.6   |    | 85.0 85.3   |    |
|                | GAE             | 77.0 77.7   |    | 77.3 78.2   |    | 80.3 79.6   |    |
| Heterogeneous  | PTE             | 69.5 63.8   |    | 70.1 64.8   |    | 69.1 64.7   |    |
|                | Metapath2Vec    | 78.5 73.8   |    | 80.6 75.8   |    | 81.9 79.7   |    |
| Multi-View     | Deepwalk-con    | 78.9 78.0   |    | 79.8 78.9   |    | 84.7 83.1   |    |
|                | MultiNMF        | 80.3 80.2   |    | 81.9 82.3   |    | 82.2 82.8   |    |
|                | MVE             | 82.0 82.4   |    | 83.0 82.8   |    | 83.4 83.1   |    |
|                | MNE             | 82.3 82.7   |    | 83.3 83.5   |    | 84.1 84.6   |    |
|                | MTNE-C          | 52.4 53.0   |    | 62.3 62.9   |    | 66.1 65.8   |    |
|                | RGAE            | 82.7 83.2   | 85.5 85.2 | 86.3 85.9   |    |              |    |

Link Prediction We select the YouTube dataset to verify the performance of link prediction. Table 4 shows that the RGAE model significantly outperforms all baseline methods. The results verify again that RGAE indeed can preserve the abundant information in multi-view networks. It is noticeable that the SDNE even outperforms all multi-view and heterogeneous network embedding approaches. By designing two kinds of graph auto-encoders, RGAE utilizes both consistent and unique information from multiple views to describe the node proximity in a detailed way, which achieves better performance than SDNE. As a result, we conclude that RGAE is able to explore the structural properties of multi-view networks.
Network Visualization We project the embeddings of AMiner dataset onto 2d vectors with t-SNE [20]. Fig. 3 shows the network visualizations of the RGAE model as well as each view’s visualization obtained by its shared and private encoders. The difference between RGAE model and the single-view model is that single-view model lacks not only the constraints of the loss function to divide the consistent information and unique information in a view, but also the cooperation and supplementary between different views. In order to make the visualization results more clear and legible, we select three from the eight categories of nodes for visualization, and each color represents a research field. We can see that our multi-view based approach works better than learning each view individually. Citation view may achieve relatively good representation effect, but there are still a few nodes that have not been assigned to the correct cluster. Therefore, it still needs more useful information from other views to complement and properly correct it to get a robust representation. The visualization of RGAE, by contrast, separates the three research fields clearly, which illustrates the necessity of modeling heterogeneous edges with consideration of all types of relationships.

4.3 Influence of Loss Functions

The visualization results have proven the importance of both consistent and unique information. In this part, we research the effect of the loss functions. In our RGAE model, there exist two loss functions, i.e. the similarity loss and the difference loss, that regularize the processes of extracting the consistent and unique information respectively. To evaluate the influences of the two loss functions, we remove similarity loss, difference loss, and both of them respectively, and show the performance in Fig. 4. The histogram clearly shows the importance of the two loss functions for our RGAE model. When we remove the similarity loss function, there is slight decline in performance. Because without similarity loss function, the quality of consistent information will be affected. Whereas there is relatively little consistent information among the views, and the proportion of the dimensions of the common representation in the final representation is small, so that the performance declination will not be quite severe. When there is no difference loss, there will be a noticeable decrease in performance, because the isolation between different view’s specific information becomes worse without the regularization of difference loss. Moreover, if we remove the similarity loss and difference loss simultaneously, the performance of the RGAE model declines further dramatically. All
these observations can demonstrate the necessity of the similarity loss and difference loss, but the degree of influence varies between the two losses.

Fig. 4. The effectiveness of the similarity loss and difference loss in our RGAE model

### 4.4 Parameter Sensitivity

With results presented in Fig. 5, we focus on the parameter sensitivity of RGAE model, including the number of embedding dimensions, $\alpha$, $\beta$, and $\gamma$. We perform node classification on AMiner dataset and link prediction on YouTube dataset to evaluate the parameter sensitivity. To explore the contributions of these parameters, we fix others to evaluate the effect of one parameter at a time on the experimental results.

Overall, different datasets and tasks have different sensitivities to embedding dimensions. On AMiner dataset, the performance increases with the dimension increasing then stabilizes when the dimension reaches 64. While on the YouTube dataset, the model performs well when the dimension is 32 and the performance decreases slightly when the dimension continues to increase. Compared with the AMiner dataset, the Youtube dataset achieves good results in lower dimensions. When the proportion of the training data set is small, a large number of dimensions tend to cause overfitting.

The curves of the experimental metrics with the parameters $\alpha$ or $\beta$ are not monotonic. Their overall trends are both first rising and then falling. Because when the proportion of similarity loss function and difference loss function are too large, the proportion of reconstruction loss will be weakened, which will affect the representation abilities of graph auto-encoders. As for $\gamma$, we find that it actually influences the results for both tasks. As we can see, it is more suitable to set the value of $\gamma$ larger than 5.

### 5 Related Work

**Network Embedding:** Network embedding is dedicated to mapping nodes in a network into a low-dimensional vector space for preserving structural information. Earlier studies such as Deepwalk [22], node2vec [8], and Struc2vec [25] use skip-gram model to preserve network structures through neighborhood sampling. Traditional deep neural networks also get widespread attention because of its nonlinear underlying structure. SDNE [34], SiNE [35], and Deepcas [14] have a strong advantage in retaining the highly nonlinear structure of the network. More recent methods adopt graph neural network to perform convolutional operations on graphs. GCN [11], GATs [33], and
GraphSAGE [9] are all representative works as end-to-end approaches for network representation learning. These studies are directed at homogeneous networks. Heterogeneous network embedding has also attracted attention because of its practical significance. PTE [29] is an extension method of LINE on heterogeneous networks. Besides, Metapath2vec [4], HIN2vec [7], and RHINE [18] use meta path to capture the structure and semantic information in heterogeneous networks.

Multi-view Learning: Another related work is about multi-view learning. Some traditional multi-view learning algorithms, such as co-training [13], co-clustering [39], and cross-domain fusion [6] analyze multi-view networks for specific tasks. MVE [24], MINES [19], MVNE [28], and mvn2vec [27] account for the first-order collaboration to align the representations of each node across views. For these studies, the models responsible for learning the network representation of each view are shallow so that they cannot capture the high-order non-linear network structure. With that in mind, we consider using deep neural networks to replace the shallow models as the basic components to embed the network. ACMVL [18] uses multiple auto-encoders to learn the specific features of each view and map all specific features to the same potential space. But it requires a supervised network to help the auto-encoder optimize its parameters. Compared with it, our model is totally unsupervised to solve the multi-view network embedding problem.

6 Conclusion

In this paper, we explore how to model the heterogeneity of edges by solving a multi-view network embedding problem and propose a novel RGAE model. More specifically, our model makes use of two types of graph auto-encoders to extract consistent and unique information of views respectively, and innovatively proposes two loss functions to distinguish these two types of information. Experimental results not only indicate the superiority of the proposed model but also investigate the contributions of two loss functions. In the future, we plan to apply the framework to more applications. A
meaningful direction is to use multi-view learning to represent general heterogeneous networks, that is, the nodes and edges of the network have multiple types at the same time.

Acknowledgement

The research was supported by National Natural Science Foundation of China (No. 61802140) and Hubei Provincial Natural Science Foundation (No. 2018CFB200).

References

1. Cai, X., Nie, F., Huang, H.: Multi-view k-means clustering on big data. In: Proceedings of the Twenty-Third International Joint conference on artificial intelligence (2013)
2. Cao, S., Lu, W., Xu, Q.: Grarep: Learning graph representations with global structural information. In: Proceedings of the 24th ACM international on conference on information and knowledge management. pp. 891–900. ACM (2015)
3. Defferrard, M., Bresson, X., Vandergheynst, P.: Convolutional neural networks on graphs with fast localized spectral filtering. In: Proceedings of the 30th Advances in neural information processing systems. pp. 3844–3852 (2016)
4. Dong, Y., Chawla, N.V., Swami, A.: metapath2vec: Scalable representation learning for heterogeneous networks. In: Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining. pp. 135–144. ACM (2017)
5. Franceschini, A., Szklarczyk, D., Frankild, S., Kuhn, M., Simonovic, M., Roth, A., Lin, J., Minguez, P., Bork, P., Von Mering, C., et al.: String v9. 1: protein-protein interaction networks, with increased coverage and integration. Nucleic acids research 41(D1), D808–D815 (2012)
6. Franco, J.S., Boyer, E.: Fusion of multiview silhouette cues using a space occupancy grid. In: Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV’05) Volume 1. vol. 2, pp. 1747–1753. IEEE (2005)
7. Fu, T.Y., Lee, W.C., Lei, Z.: Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management. pp. 1797–1806. ACM (2017)
8. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 855–864. ACM (2016)
9. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: Advances in Neural Information Processing Systems. pp. 1024–1034 (2017)
10. He, K., Sun, Y., Bindel, D., Hopcroft, J., Li, Y.: Detecting overlapping communities from local spectral subspaces. In: Proceedings of the 2015 IEEE International Conference on Data Mining. pp. 769–774. IEEE (2015)
11. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
12. Kipf, T.N., Welling, M.: Variational graph auto-encoders. arXiv preprint arXiv:1611.07308 (2016)
13. Kumar, A., Daumé, H.: A co-training approach for multi-view spectral clustering. In: Proceedings of the 28th International Conference on Machine Learning (ICML-11). pp. 393–400 (2011)
14. Li, C., Ma, J., Guo, X., Mei, Q.: Deepcas: An end-to-end predictor of information cascades. In: Proceedings of the 26th international conference on World Wide Web. pp. 577–586 (2017)
15. Li, J., Wu, L., Liu, H.: Multi-level network embedding with boosted low-rank matrix approximation. arXiv preprint arXiv:1808.08627 (2018)
16. Li, S.Z., Zhu, L., Zhang, Z., Blake, A., Zhang, H., Shum, H.: Statistical learning of multi-view face detection. In: Proceedings of the European Conference on Computer Vision. pp. 67–81. Springer (2002)
17. Liu, J., Wang, C., Gao, J., Han, J.: Multi-view clustering via joint nonnegative matrix factorization. In: Proceedings of the 2013 SIAM International Conference on Data Mining. pp. 252–260. SIAM (2013)
18. Lu, R.k., Liu, J.w., Wang, Y.f., Xie, H.j., Zuo, X.: Auto-encoder based co-training multi-view representation learning. In: Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining. pp. 119–130. Springer (2019)
19. Ma, Y., Ren, Z., Jiang, Z., Tang, J., Yin, D.: Multi-dimensional network embedding with hierarchical structure. In: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining. pp. 387–395. ACM (2018)
20. Maaten, L.v.d., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research 9(Nov), 2579–2605 (2008)
21. Pan, Y., Yao, T., Mei, T., Li, H., Ngo, C.W., Rui, Y.: Click-through-based cross-view learning for image search. In: Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval. pp. 717–726. ACM (2014)
22. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: Online learning of social representations. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 701–710. ACM (2014)
23. Qiu, J., Dong, Y., Ma, H., Li, J., Wang, K., Tang, J.: Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec. In: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining. pp. 459–467 (2018)
24. Qu, M., Tang, J., Shang, J., Ren, X., Zhang, M., Han, J.: An attention-based collaboration framework for multi-view network representation learning. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management. pp. 1767–1776. ACM (2017)
25. Ribeiro, L.F., Saverese, P.H., Figueiredo, D.R.: struc2vec: Learning node representations from structural identity. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 385–394. ACM (2017)
26. Salakhutdinov, R., Hinton, G.: Semantic hashing. International Journal of Approximate Reasoning 50(7), 969–978 (2009)
27. Shi, Y., Han, F., He, X., He, X., Yang, C., Luo, J., Han, J., Han, J.: mvn2vec: Preservation and collaboration in multi-view network embedding. arXiv preprint arXiv:1801.06597 (2018)
28. Sun, Y., Bui, N., Hsieh, T.Y., Honavar, V.: Multi-view network embedding via graph factorization clustering and co-regularized multi-view agreement. In: Proceedings of the 2018 IEEE International Conference on Data Mining Workshops (ICDMW). pp. 1006–1013. IEEE (2018)
29. Tang, J., Qu, M., Mei, Q.: Pte: Predictive text embedding through large-scale heterogeneous text networks. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 1165–1174. ACM (2015)
30. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: Line: Large-scale information network embedding. Proceedings of the 24th International Conference on World Wide Web (2015)
31. Tang, J., Zhang, J., Yao, L., Li, J., Zhang, L., Su, Z.: Arnetminer: extraction and mining of academic social networks. In: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 990–998. ACM (2008)
32. Tang, L., Liu, H.: Relational learning via latent social dimensions. In: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 817–826. ACM (2009)
33. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
34. Wang, D., Cui, P., Zhu, W.: Structural deep network embedding. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 1225–1234. ACM (2016)
35. Wang, S., Tang, J., Aggarwal, C., Chang, Y., Liu, H.: Signed network embedding in social media. In: Proceedings of the 2017 SIAM international conference on data mining. pp. 327–335. SIAM (2017)
36. Wang, S., Tang, J., Aggarwal, C., Liu, H.: Linked document embedding for classification. In: Proceedings of the 25th ACM international on conference on information and knowledge management. pp. 115–124. ACM (2016)
37. Xu, L., Wei, X., Cao, J., Philip, S.Y.: Multi-task network embedding. International Journal of Data Science and Analytics 8(2), 183–198 (2019)
38. Yang, J., Leskovec, J.: Defining and evaluating network communities based on ground-truth. Knowledge and Information Systems 42(1), 181–213 (2015)
39. Yao, X., Han, J., Zhang, D., Nie, F.: Revisiting co-saliency detection: A novel approach based on two-stage multi-view spectral rotation co-clustering. IEEE Transactions on Image Processing 26(7), 3196–3209 (2017)
40. Zhang, H., Qiu, L., Yi, L., Song, Y.: Scalable multiplex network embedding. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence. vol. 18, pp. 3082–3088 (2018)