Adversarial network embedding on heterogeneous information networks

Ting Lan, Changxuan Wu, Chunyan Yu, Xiu Wang
College of Mathematics and Computer Science, Fuzhou University, Fuzhou 350108, China
wangx@fzu.edu.cn

Abstract. Network embedding has been proven to be helpful for solving real-world problems. Moreover, real-world networks are often heterogeneous information networks (HINs). In this paper, we propose a new adversarial framework for heterogeneous network embedding, namely AGNE-HIN. AGNE-HIN can learn latent code distribution in the network through a generative adversarial way. What's more, to reduce the global smoothness of the embedded vector caused by GAN, we apply perturbation to the input to form adversarial data. Experimental results verify our design and demonstrate the effectiveness of the proposed method in node clustering, link prediction and similarity ranking tasks.

1. Introduction
Real-world networks are often heterogeneous information networks (HINs)[1], which have various types of nodes and relations between nodes. Modeling and analyzing these networks play a significant role in many network analysis tasks, including clustering[2], link prediction[3] and so on[4][5]. However, due to the high dimensionality and the sparseness of network data, traditional graph representation methods are difficult to be applied in the real scene, and the emergence of network embedding has alleviated this problem.

Network embedding[6]-[8] aims to represent nodes in the network as the low-dimensional, dense and real-valued vectors. It has transformed the task of learning complex models for the network into learning the robust representation of nodes in the network.

The heterogeneous network representation learning methods[9]-[11] are mainly focus on randomly selecting nodes from the network as negative samples and have the problem of over-smoothing, these can easily lead to poor representation in dealing with real-world networks[12]. Some adversarial-based frameworks(such as HeGAN)[13][14] represent nodes in HIN through a generative adversarial way, and learn the potential distribution of nodes. However, they are inefficient to embed large networks and they have more complex input structures. For example, HeGAN adds different particular relationships as constraints. So we mainly use simpler structure to generate more robust node representations.

In this paper, we propose a novel adversarial framework to generate node representations, namely AGNE-HIN. AGNE-HIN integrate perturbation into the adversarial model to form adversarial data, which extends the original clean data, what’more, instead of using the constraints of particular relationships, AGNE-HIN set the L2 norm constraint according to the connectivity pattern of the node pair. The loss of the generator and discriminator take into account the adversarial data so that our generator can generate more representative and robust node representations.
2. Symbols and Problem Definition

**Heterogeneous information networks.** A HIN $\zeta = \{V, E, \mathcal{S}, \mathcal{R}, \rho, \sigma\}$ is a graph, where $\mathcal{S}$ and $\mathcal{R}$ represent the types of nodes and edges, respectively. $\rho$ is a node type mapping function: $V \rightarrow \mathcal{S}$, $\sigma$ is an edge type mapping function: $E \rightarrow \mathcal{R}$, $V = \{v_i\}$ represents the node set, $e_{i,j} = \langle v_i, v_j \rangle \in E$ is the edge between node $v_i$ and node $v_j$.

**Heterogeneous information network embedding.** HIN embedding aims to learning a real vector $d_\mathcal{R}$ for each node $v_i \in V$ in the HIN, where $d \ll |V|$. The learned real value vector should be able to preserve the correlation between the nodes in the HIN.

3. Model

In this paper, we employ GAN[15] to learn node representations in HINs. As shown in Fig.1, our model mainly consists of two main parts: generator and discriminator. The discriminator and the generator compete with each other to achieve their best performance.

![Fig.1 The architecture of our model](image)

3.1. Model details

Our generator samples nodes from a Gaussian distribution, and the generated fake nodes should be close to the nodes of the real data to confuse the discriminator. Therefore, given a node $u \in V$, $u$ and $v \in V$ connected, the generator tries to generate a fake node $v'$ connected to the node $u$. In other words, the representation of the fake node should be as similar as possible to the real node $v$.

The goal of the discriminator is to determine whether a sample node is a node that is actually connected to a given input node. Our discriminator can be expressed as follows:

$$
D(e_u | u) = \frac{1}{1 + \exp(-e_u^D e_w^T)}
$$

Where $w$ is a sample node (possibly a fake or true node), $e_u^D \in \mathbb{R}^{v \times d}$ is the embedding of a node $u$, and $e_w \in \mathbb{R}^{v \times d}$ is the embedding of the sample node $w$.

We define the adversarial perturbation $\eta_{adv}$ in the embedding space and set the L2 norm constraint according to the connectivity pattern of the node pair. Then the adversarial perturbation is added into the input to form an adversarial example, which extends the original clean data, and then we use the new mixed data to train our model. Before each training, first, we solve the optimization problem to generate the adversarial perturbation $\eta_{adv}$ corresponding to the input $x$, so that it has anti-disturbance to the current model. What's more, because of its computational complexity, we use fast gradient descent to obtain the adversarial perturbation. During the training process, the embedded vector will perform stochastic gradient descent[14] optimization according to the designed perturbation.
3.2. Loss
The generator hopes that after the perturbation is added to the input, the embedding generated by itself can still confuse the discriminator. In addition, we add smoothness constraints to make the fake node's embedding $e'_v$ and the embedding of the given node $u$ have similar representations. So, the loss of our generator is shown below:

$$L^G = E_{u,v',c'\sim G(u)} - \log D(e'_v | u, \eta_{adv}) + \lambda^G \|e'_v - e'_u\|^2$$

Where $\lambda^G > 0$ controls the regularization term.

For the discriminator, the loss of our discriminator is as follows:

$$L^D = L^D_{true} + L^D_{fake} + \lambda^D \|\rho^D\|_2^2$$

Where $\lambda^D > 0$ controls the regularization term to prevent overfitting. $\theta^D$ is the parameter of the discriminator.

The node $u$ and the node $v$ are indeed connected in the HIN. The discriminator hopes to get high scores for the node $v$:

$$L^D_{true} = E_{(u,v)\sim P_{true}} - \log D(e_u | u)$$

When a node $u$ is connected to a node $v$, but $v$ is not in the HIN, then $v$ is a fake node generated from the generator, in which case the discriminator wishes to get low scores for the node $v$:

$$L^D_{fake} = E_{u,v',c'\sim G(u)} - \log (1 - D(e'_v | u))$$

Where $e'_v$ is the embedding of the fake node $v$.

4. Experiments
4.1. Experimental Setups
In our experiments, we brief the DBLP dataset to construct the HIN. The criteria are that it cannot be too small and too sparse. In other words, it should contain at least one edge per node per relation type. Therefore, the HIN used in our experiments contains 31975 nodes which have a total of three node types: authors, papers and conferences. The complete DBLP dataset is available online. Besides, the HIN has 627974 edges, including 473480 paper-author edges and 154494 paper-conference edges.

For AGNE-HIN's training, in each iteration, we first fix the parameters in the generator $G$ and generator fake nodes from $G$ to update the parameters of the discriminator, so as to improve the discriminator. After updating the discriminator, we fix the parameters of the discriminator to update the generator. We do these steps many times until our model converges.

4.2. Experiment Analysis
Node clustering
The node clustering experiment is to aggregate all nodes into several different classes according to the characteristics of different node types. Therefore, we use it to evaluate the ability of our model to learn the characteristics of nodes. In this task, we classify the nodes of the paper and use the field associated with the node as the label of the node, so that the learned classifier can determine the category of each node. Among them, the paper nodes have seven categories. We use the K-means algorithm to cluster the data and evaluate the clustering results based on normalized mutual information (NMI). And, a higher value represents better node representations. All clustering results are conducted 10 times and the average performance is reported in Table 1.

---

1 http://gdm.fudan.edu.cn/GDMWiki/Wiki.jsp?page=Network\%20Data\Set
2 0. Distributed Parallel Computing, 1. World Wide Web, 2. Algorithms Theory, 3. Databases, 4. Information Retrieval, 5. Data Mining, 6. Scientific Computing
Table 1 Clustering Results

| methods     | papers |
|-------------|--------|
| Metapath2vec | 19.63  |
| HeGAN       | 22.08  |
| AGNE-HIN    | 23.85  |

From Table 1, we can conclude that, with respect to the above metric, the proposed AGNE-HIN model achieves the best performance among all methods, demonstrating the effectiveness of AGNE-HIN in learning the node representations of HIN. The results between metapath2vec and GAN-based embedding methods (HeGAN and AGNE-HIN) has proved the superiority of the adversarial principle. AGNE-HIN outperforms HeGAN, showing the effectiveness and importance of perturbations when learning robust node representations.

**Link Prediction**

Link prediction\[16\] is used to predict the possible links between two types of nodes, so we use it to evaluate our model's performance to learn the relationship between two types of nodes. In this task, we predict the paper-author links in DBLP. We randomly hide 20% of such links from the original network and use them as a test set to verify performance. Finally, the remaining network is used to learn node representations.

Following the commonly used evaluation criteria in similar tasks, we use Accuracy as the evaluation criteria in our experiment. A higher value indicates better performance. We performed ten experiments and used the average as the final score.

Table 2 Link Prediction Results

| methods     | Acc     |
|-------------|---------|
| Metapath2vec | 47.15   |
| HeGAN       | 49.23   |
| AGNE-HIN    | 50.68   |

The details of our experimental results on link prediction are shown in Table 2. The results show that AGNE-HIN achieves better performance in all methods. For example, AGNE-HIN has increased the accuracy from 2.9% compared with HeGAN to 7.5% compared with metapath2vec. We integrate effective perturbations into GAN to learn the potential relationships between different types of nodes so that we can predict the unseen and non-existent links.

**Similarity Ranking**

We perform the similarity ranking experiment to verify the effectiveness of the proposed AGNE-HIN. This experiment can illustrate the similarity between nodes of the same type.

First of all, we select three conferences from all the conferences, namely "FOCS", "TREC" and "ER", and we select the top 10 conferences that are most relevant to the three conferences as our benchmark for this task. These data are shown in Table 3. It should be noted that the correlations between conferences are determined by the number of co-authors between them, and the greater the number of co-authors between conferences, the stronger their correlations.

Table 3 Benchmark

| Rank | FOCS | TREC | ER   |
|------|------|------|------|
| 1    | STOC | SIGIR| ICDE |
| 2    | SODA | CIKM | CAISE|
| 3    | ICALP| ECIR | DEXA |
| 4    | ESA  | WWW  | CIKM |
| 5    | STACS| CLEF | SIGMOD|
| 6    | SPAA | RIAO | VLDB |
| 7    | COCO | AIRS | DASFAA|
Then, we select "FOCS", "TREC" and "ER" as query nodes, and we use cosine similarity to calculate the correlations (distance) between the query nodes and their corresponding nodes in Table 3. In the end, their similar results corresponding to the query nodes are listed in Table 4.

### Table 4 Similarity Ranking Results

| Rank | FOCS | TREC | ER |
|------|------|------|----|
|      | M2vec | HeGAN | Ours | M2vec | HeGAN | Ours | M2vec | HeGAN | Ours |
| 1    | STOC  | STOC  | SIGIR | SIGIR | SIGIR | CAISE | CAISE | CAISE |
| 2    | SODA  | SODA  | ECIR  | ECIR  | ECIR  | VLDB  | VLDB  | DEXA |
| 3    | ICALP | ICALP | CLEF  | CIKM  | CIKM  | ICDE  | DEXA  | ICDE |
| 4    | COCO  | COPO  | SPAA  | CLEF  | CLEF  | DASFAA| ICDE  | VLDB |
| 5    | SPAA  | PODC  | COCO  | AIRS  | AIRS  | DEXA  | SIGMOD| DASFAA|
| 6    | PODC  | SPAA  | PODC  | RIAO  | RIAO  | RIAO  | SIGMOD| DASFAA|
| 7    | ESA   | ESA   | APPROX| WWW   | JCDL  | WWW   | WISE  | CIKM |
| 8    | APPROX| APPROX| ESA   | JCDL  | WWW   | JCDL  | CIKM  | WISE |
| 9    | ISAAC | STACS | STACS | ICDM  | ICDM  | KDD   | OTM   | OTM |
| 10   | STACS | ISAAC | ISAAC | KDD   | KDD   | ICDM  | ICEIS | ICEIS|

From Table 4, we can know that for the node "ER'', there are up to four conferences ("CAISE'', "DEXA'', "CIKM'' and "ICEIS'') in Metapath2vec in the same relative orders as the benchmark. In HeGAN, there are at most five conferences ("CAISE'', "DEXA'', "SIGMOD'', "DASFAA'' and "ICEIS'') in the same relative orders as the benchmark. In the proposed AGNE-HIN, there are at most five conferences ("CAISE'', "DEXA'', "VLDB'', "DASFAA'' and "ICEIS'') in the same relative orders as the benchmark. Other nodes do the same analysis. Their relative orders are the bold font in Table 4. What’s more, for these query nodes, the first three conferences obtained by the proposed AGNE-HIN are the same as the first three conferences of the corresponding node in the benchmark, although they are in a different order.

The proposed AGNE-HIN can learn the features between different node types, and for the same type of nodes, AGNE-HIN is able to capture the commonness between nodes, so as to obtain high-quality node representations.

### 5. Conclusion

In this paper, we propose a novel adversarially regularized framework to generate high-quality node representations, namely AGNE-HIN. AGNE-HIN obtains the potential data distribution through a generative adversarial way. Moreover, we integrate perturbation into our model to form adversarial data, which can reduce the global smoothness of the embedded vectors caused by GAN. Through the above operations, we can achieve model's robustness and better generalization performance. We test our model on three tasks: node classification, link prediction and similarity ranking. Experimental results show that our model is better than baselines.

### Acknowledgments

This work was supported by the Fujian Foundation of Science 2018J01794 and the Fujian Health-Education Joint Project WKJ2016-2-26.

### References

[1] C. Shi, Y. Li, J. Zhang, Y. Sun, and S. Y. Philip, “A survey of heterogeneous information network analysis,” IEEE Transactions on Knowledge and Data Engineering, vol. 29, no. 1,
pp. 17–37, 2016.

[2] Y. Sun, Y. Yu, and J. Han, “Ranking-based clustering of heterogeneous information networks with star network schema,” in Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2009, pp. 797–806.

[3] D. Davis, R. Lichtenwalter, and N. V. Chawla, “Multi-relational link prediction in heterogeneous information networks,” in 2011 International Conference on Advances in Social Networks Analysis and Mining. IEEE, 2011, pp. 281–288.

[4] C. Shi, J. Liu, F. Zhuang, S. Y. Philip, and B. Wu, “Integrating heterogeneous information via flexible regularization framework for recommendation,” Knowledge and Information Systems, vol. 49, no. 3, pp. 835–859, 2016.

[5] Q. Zhan, J. Zhang, S. Wang, S. Y. Philip, and J. Xie, “Influence maximization across partially aligned heterogeneous social networks,” in Pacific-Asia Conference on Knowledge Discovery and Data Mining. Springer, 2015, pp. 58–69.

[6] P. Goyal and E. Ferrara, “Graph embedding techniques, applications, and performance: A survey,” Knowledge-Based Systems, vol. 151, pp. 78–94, 2018.

[7] H. Cai, V. W. Zheng, and K. C.-C. Chang, “A comprehensive survey of graph embedding: Problems, techniques, and applications,” IEEE Transactions on Knowledge and Data Engineering, vol. 30, no. 9, pp. 1616–1637, 2018.

[8] P. Cui, X. Wang, J. Pei, and W. Zhu, “A survey on network embedding,” IEEE Transactions on Knowledge and Data Engineering, vol. 31, no. 5, pp. 833–852, 2018.

[9] Y. Dong, N. V. Chawla, and A. Swami, “metapath2vec: Scalable representation learning for heterogeneous networks,” in Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining. ACM, 2017, pp. 135–144.

[10] T.-y. Fu, W.-C. Lee, and Z. Lei, “Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning,” in Proceedings of the 2017 ACM on Conference on Information and Knowledge Management. ACM, 2017, pp. 1797–1806.

[11] Z. Huang and N. Mamoulis, “Heterogeneous information network embedding for meta path based proximity,” arXiv preprint arXiv:1701.05291, 2017.

[12] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, “Adversarial autoencoders,” arXiv preprint arXiv:1511.05644, 2015.

[13] Yu W, Zheng C, Cheng W, et al. Learning deep network representations with adversarially regularized autoencoders[C]//Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 2018: 2663-2671.

[14] He D, Zhai L, Li Z, et al. Adversarial Mutual Information Learning for Network Embedding[J].

[15] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in Advances in neural information processing systems, 2014, pp. 2672–2680.

[16] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” arXiv preprint arXiv:1412.6572, 2014.

[17] Sun Y, Han J, Yan X, et al. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks[J]. Proceedings of the VLDB Endowment, 2011, 4(11): 992-1003.

[18] Dong Y, Zhang J, Tang J, et al. Coupledlp: Link prediction in coupled networks[C]//Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2015: 199-208.