A View-Adversarial Framework for Multi-View Network Embedding

Dongqi Fu*, Zhe Xu*, Bo Li, Hanghang Tong, and Jingrui He
University of Illinois at Urbana-Champaign
{dongqif2, zhexu3, lbo, htong, jingrui}@illinois.edu

ABSTRACT

Network embedding has demonstrated effective empirical performance for various network mining tasks such as node classification, link prediction, clustering, and anomaly detection. However, most of these algorithms focus on the single-view network scenario. From a real-world perspective, one individual node can have different connectivity patterns in different networks. For example, one user can have different relationships on Twitter, Facebook, and LinkedIn due to varying user behaviors on different platforms. In this case, jointly considering the structural information from multiple platforms (i.e., multiple views) can potentially lead to more comprehensive node representations, and eliminate noises and bias from a single view. In this paper, we propose a view-adversarial framework to generate comprehensive and robust multi-view network representations named VANE, which is based on two adversarial games. The first adversarial game enhances the comprehensiveness of the node representation by discriminating the view information which is obtained from the subgraph induced by neighbors of that node. The second adversarial game improves the robustness of the node representation with the challenging of fake node representations from the generative adversarial net. We conduct extensive experiments on downstream tasks with real-world multi-view networks, which shows that our proposed VANE framework significantly outperforms other baseline methods.

CCS CONCEPTS

• **Mathematics of computing → Graph algorithms**; • **Computing methodologies → Learning latent representations**;

KEYWORDS

Network Embedding; Multi-View Network; Adversarial Learning

ACM Reference Format:

Dongqi Fu*, Zhe Xu*, Bo Li, Hanghang Tong, and Jingrui He. 2020. A View-Adversarial Framework for Multi-View Network Embedding. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM '20), October 19–23, 2020, Virtual Event, Ireland*. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3340531.3412127

*Both authors contributed equally to this research.

CIKM '20, October 19–23, 2020, Virtual Event, Ireland

ACM ISBN 978-1-4503-6859-9/20/10…$15.00
https://doi.org/10.1145/3340531.3412127

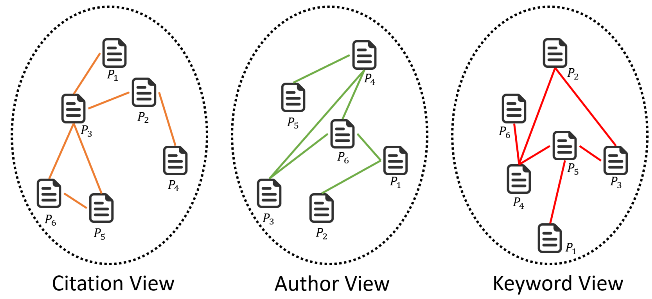

Figure 1: An illustrative multi-view network example of 3 views with 6 papers. In the citation view, an edge connects two papers if one paper cites the other. In the author view, an edge exists if two papers share at least one common author. In the keyword view, an edge exists if two papers share at least one common keyword.

1 INTRODUCTION

Network embedding algorithms provide network representations for many graph mining tasks such as node classification, link prediction, clustering, and anomaly detection. Classical network embedding algorithms such as DeepWalk [7], LINE [10] and node2vec [5] capture topology information like local neighborhood connectivity pattern, structural role, and other high-order proximities to represent each node. Moreover, in order to obtain robust representations, many network embedding algorithms have been proposed by leveraging the principle of the generative adversarial net (GAN) [3], such as ANE [2] and GraphGAN [12]. To this end, the generative model tries to fit the underlying connectivity distribution of the network and then produces fake samples (i.e., fake nodes, fake relations or fake representations) to fool the discriminative model, while the discriminative model tries to distinguish the generated fake samples from the ground truth samples.

Traditional network embedding algorithms like [5, 7, 10] and state-of-the-art GAN-based network embedding algorithms like [2, 12] only focus on the single network scheme. In the multi-view scenario (e.g., Figure 1), the same set of nodes can have different connectivity patterns in different views. Take the scientific papers for example, in the view of citations, two papers are connected with an edge if there is a reference record. However, in the common-author view, these two papers are not connected if they do not share at least one common author. Also, in the time-evolving graph, each graph snapshot can be regarded as a view. Jointly embedding the structural information from multiple views can lead to a comprehensive node representation and remove noise and bias from a single view [8, 14]. Recently, with the advances of graph neural networks, many multi-view network embedding frameworks [1, 9] have been proposed to obtain the attributed node representation.

Most, if not all, existing multi-view network embedding algorithms follow the nature of the view-collaboration mechanism [8] to integrate specific node representations from each specific view, in order to generate the comprehensive node representations. In this paper, we propose a view-adversarial framework for multi-view network embedding named VANE, which consists of two adversarial games. The first adversarial game ensures the comprehensiveness of node representations: the feature extractor aims to extract comprehensive node representations containing view-invariant structural information to fool the view discriminator, while the view discriminator aims to distinguish which node representation comes from which view by discriminating view-dependent structural information. The second adversarial game ensures the robustness of node representations: the generator tries to fit the distribution of the extracted node representations to generate fake node representations, and the node representation discriminator tries to discriminate fake representations from real representations; in the meanwhile, the feature extractor tries to provide robust node representations which are hard to fit.

2 PROBLEM DEFINITION

A view is defined as a single type of edge. A multi-view network with k views is defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E}_1, \ldots, \mathcal{E}_k)$, where $\mathcal{V}$ is the set of nodes, and $\mathcal{E}_i$ is the set of edges in the i-th view. Thus, we formally define the *Multi-View Network Embedding* as follows.

Problem. *Multi-View Network Embedding*

Input: *the multi-view network* $\mathcal{G} = (\mathcal{V}, \mathcal{E}_1, \ldots, \mathcal{E}_k)$.

Output: *the robust node representations* $\{\mathbf{x}_v\}_{v \in \mathcal{V}} \in \mathbb{R}^d$ *with* $d \ll |\mathcal{V}|$, *which are consistent across* k *different views.*

3 PROPOSED MODEL

In this section, we introduce the proposed VANE model from the overview to the details of two adversarial games.

3.1 Overview of the Framework - VANE

The proposed VANE framework is shown in Figure 2. It consists of two adversarial games. To obtain comprehensive node representations, the first adversarial game involves the feature extractor F and the view discriminator D_S. The feature extractor F tries to extract the view-invariant node representation across different views, while the view discriminator D_S tries to discriminate the view-dependent structural information (i.e., subgraph representation) from node representations. To obtain robust node representations, the second adversarial game involves the node embedding model F_N, the node representation generator G, and the node representation discriminator D_N. Challenged by fake node representations from the generator G, the node embedding model F_N tries to provide robust node representations for the node representation discriminator D_N.

3.2 The First Adversarial Game

The first adversarial game consists of the feature extractor F and the view discriminator D_S. In the feature extractor F, the node embedding model F_N is instantiated by the regular embedding layer which is a fully-connected layer for one-hot node vectors. The subgraph embedding model F_S is instantiated by an LSTM [6] layer to aggregate a sequence of node representation vectors into a subgraph representation vector. The view discriminator D_S is a multi-class classifier and instantiated by a multilayer perceptron to take the subgraph representation vector as the feature and the index of view where the subgraph comes from as the label.

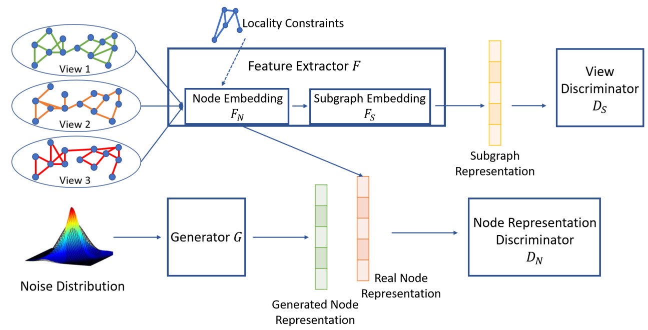

Figure 2: The framework of VANE.

We first preprocess all input views into node sequences by random walks where the node sequence S is denoted as $S = \{v_1, v_2, \ldots, \}$. Then we input node sequences into the feature extractor F individually. Constructed by F_N and F_S, the feature extractor F is a shared neural network by all node sequences. In F_N, the one-hot vector of each node v_i in the sequence S is represented by the node embedding vector $\mathbf{x}_{v_i}$. After that, the subgraph embedding model F_S aggregates the sequence of node embedding vectors into a subgraph embedding vector denoted as $F(S)$. During the first adversarial game, F tries to extract indistinguishable (i.e., view-invariant) subgraph embedding vectors to fool the view discriminator D_S, while D_S tends to distinguish the view-specific information from the subgraph representations to discriminate the source view.

Suppose we have k views in the given multi-view network. For the i-th view discrimination ($1 \le i \le k$), the distribution of subgraph representation $F(S)$ from the i-th view denotes as $F(S) \sim p_i(F(S))$, the distribution of node sequence $F(S)$ from views other than the i-th view defines as $\bar{p_i}(F(S)) = \frac{1}{k-1} \sum_{m \neq i} p_m(F(S))$, and $D_S(F(S))$ represents the probability that $F(S)$ came from the i-th view rather than other views. Thus, the objective J_S for the i-th view subgraph representation and discrimination states as follows.

$$\begin{aligned} &\min_{F} \max_{D_S} J_S(D_S, F) \\ &= \min_{F} \max_{D_S} \mathbb{E}_{F(S) \sim p_i(F(S))} [\log(D_S(F(S)))] \\ &\quad + \mathbb{E}_{F(S) \sim \bar{p_i}(F(S))} [\log(1 - D_S(F(S)))] \end{aligned} \tag{1}$$

where the feature extractor F and the view discriminator D_S aim to play the first adversarial game converging to the equilibrium as stated in [13], $p_1(F(S)) = p_2(F(S)) = \cdots = p_k(F(S))$, which suggests that the probability distribution of the subgraph representation $F(S)$ from each view equals to each other.

To further improve the ability of our model to capture the local topology information of the node sequence from each specific view, we add the cosine similarity based locality constraints on the node embedding model F_N with minimizing the following objective J_{LC}.

$$\min_{F_N} J_{LC}(F_N(v_i), F_N(v_j)) = \min_{F_N} \mathbb{E}_{v_i, v_j \in S} [1 - \cos(F_N(v_i), F_N(v_j))] \tag{2}$$

where node v_i and v_j come from the same node sequence S.

3.3 The Second Adversarial Game

To enhance the robustness of extracted node representations from the first adversarial game, the second adversarial minimax game involves three players: the node representation generator G, the node embedding model F_N, and the node representation discriminator D_N. The generator G and node representation discriminator D_N are instantiated by multilayer perceptron classifiers.

During the second game, the generator G generates fake node representations to fit the distribution of node representations produced by F_N, and F_N tries to provide robust node representations that are hard to fit to help the discriminator D_N tell the fake node embedding vectors. The objective function J_N of the second adversarial game is described as follows.

$$\begin{aligned} &\min_{G} \max_{D_N} \max_{F_N} J_N(D_N, G, F_N) \\ =&\min_{G} \max_{D_N} \max_{F_N} \mathbb{E}_{F_N(v) \sim p_{data}(F_N(v))}[\log(D_N(F_N(v)))] \\ &+ \mathbb{E}_{\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})}[\log(1 - D_N(G(\mathbf{z})))] \end{aligned} \tag{3}$$

where $F_N(v)$ is the real node representation, and $G(\mathbf{z})$ is the generated fake node representation from the noise vector $\mathbf{z}$.

3.4 Optimization

We summarize the training procedure of the VANE framework [1] in Alg. 1, where two adversarial games are executed alternatively.

In Alg. 1, we first randomly initialize the parameter θ of each component of the VANE framework. In Step 3 and Step 4, we prepare the node sequence along with the ground truth view label and generate the fake node representation. Steps 5-7 execute the first adversarial game for comprehensive node representations by updating the feature extractor F and the view discriminator D_S. Steps 8-10 execute the second adversarial game for robust node representations, where the generator G and the discriminator D_N compete with each other to force the node representation model F_N to provide robust (i.e. hard to fit) node representations. In practice, the training procedure for each data sample (S, y) is independent with each other, thus we utilize the mini-batch gradient descent to optimize the VANE framework in parallel.

4 EXPERIMENTS

4.1 Datasets

Aminer: Aminer [11][2] is an academic literature dataset, which contains 27,734 papers as nodes. We observe two types of views: (1) Citation view, where an edge represents for a reference record between two papers; (2) Common-author view, where an edge represents that two papers share at least one common author. For the mentioned two views, we have 111,819 and 525,623 edges.

Twitter-Rugby: Twitter-Rugby [4][3] is a collection of 850 rugby-related Twitter users. We observe three views: (1) Follow view, where an edge stands for a user following another user; (2) Mention view, where an edge stands for a user mentioning another user in his/her Tweet; (3) Retweet view, where an edge stands for a user retweeting another user's Tweet. For the above three views, we have 22,861, 21,660, and 9,627 edges, respectively.

[1] https://github.com/DongqiFu/VANE
[2] https://www.aminer.cn/citation
[3] http://mlg.ucd.ie/aggregation

Algorithm 1 Stochastic Training Procedure for VANE

Input:
multi-view network $\mathcal{G} = (\mathcal{V}, \mathcal{E}_1, \ldots, \mathcal{E}_k)$, r node sequences sampled from each view, max iteration T, noise vector $\mathbf{z}$.
Output:
embedding vector $\mathbf{x}_v$ for each node $v \in \mathcal{V}$.

1: **for** t in T iterations **do**
2:   **for** node sequence S in $k \times r$ shuffled sequence samples **do**
3:     Form data samples (S, y), y is the true label indicating the source view where S comes from.
4:     Use generator G to generate a fake node embedding vector $G(\mathbf{z})$ from the noise vector $\mathbf{z}$.
5:     Update the node embedding model F_N by descending its gradient w.r.t. Eq. (2) with S.
6:     Update the feature extractor F by descending its gradient w.r.t. Eq. (1) with S and y by fixing θ_{D_S}.
7:     Update the view discriminator D_S by ascending its gradient w.r.t. Eq. (1) with S and y by fixing θ_F.
8:     Update the feature extractor F_N by ascending its gradient w.r.t. Eq. (3) by fixing θ_{D_N}.
9:     Update the generator G by descending its gradient w.r.t. Eq. (3) with $G(\mathbf{z})$ by fixing θ_{D_N}.
10:     Update the node representation discriminator D_N by ascending its gradient w.r.t. Eq. (3) with $G(\mathbf{z})$ and $\mathbf{x}_v$ (i.e., $F_N(S)$) by fixing θ_F and θ_G.
11:   **end for**
12: **end for**

4.2 Baseline Methods

We extract node sequences by the random walk [7] and the biased random walk [5], such that two versions of VANE are named as VANE-RW and VANE-BRW. We compare VANE with single-view methods like DeepWalk [7], node2vec [5] and GraphGAN [12], and multi-view methods like MNE [14] and MVE [8]. For single-view algorithms, we combine all individual views into a combined view, where the edge exists if it exists in any specific view. Moreover, MNE is designed to generate node embeddings for each view.

4.3 Effectiveness Comparison

Node Classification. In the Aminer, we adopt venues as node labels and node embeddings as node features. We shuffle the dataset and sample 90% as the training set for a k-NN classifier and test on the rest 10%. We train 10 times and report the mean and standard deviation of accuracy in Table 1. Since MNE cannot get embedding results on an Intel i7 CPU, 64GB RAM machine within 96 hours, we don't report it. In the Twitter-Rugby, we adopt node embeddings as node features and user geo-locations as node labels. We use the same classifier setting in the Aminer and report results in Table 2.

Link Prediction. For each dataset, we remove 10% of the shared common links (i.e., the edge exists in every view) and learn node embeddings on the truncated graph. We denote removed links as the positive samples, and the links never appeared as negative samples. We sample the same number of negative samples and positive samples and use the cosine similarity to measure the proximity between two nodes. In the Aminer, there are many missing links between research communities. To generate quality negative samples, we conduct link prediction tasks only on Bioinformatics venue papers. In the Twitter-Rugby, since all users share the characteristic of rugby enthusiast, we use the whole dataset. The link prediction performances on two datasets are shown in Table 1 and Table 2.

Methods	View	Accuracy (%)	
		Node Classification	Link Prediction
DeepWalk	Citation	78.03±0.72	95.99
	Common-Author	72.72±0.77	96.29
	Combined	74.94±0.54	97.28
node2vec	Citation	78.05±0.57	97.73
	Common-Author	73.69±0.68	95.58
	Combined	74.88±0.91	97.85
GraphGAN	Citation	74.29±1.20	88.93
	Common-Author	72.07±1.11	89.57
	Combined	71.69±1.06	90.21
MNE	Citation	N/A	54.25
	Common-Author	N/A	52.32
MVE	All	80.16±0.42	72.82
VANE-RW	All	78.84±0.63	97.62
VANE-BRW	All	**80.79±0.80**	**98.53**

Table 1: Performance on the Aminer Dataset

Methods	View	Accuracy (%)	
		Node Classification	Link Prediction
DeepWalk	Follow	70.95±2.56	50.30
	Mention	69.64±5.46	50.27
	Retweet	73.78±5.18	52.24
	Combined	66.47±2.85	50.03
node2vec	Follow	79.52±4.42	65.45
	Mention	79.64±3.47	62.94
	Retweet	81.83±4.31	52.18
	Combined	80.59±2.75	60.61
GraphGAN	Follow	76.15±1.92	53.97
	Mention	71.95±2.74	51.88
	Retweet	39.20±2.42	50.21
	Combined	72.44±1.69	55.41
MNE	Follow	85.66±2.87	56.37
	Mention	84.70±3.45	74.66
	Retweet	85.06±3.42	76.15
MVE	All	83.76±4.90	68.85
VANE-RW	All	82.89±2.38	69.40
VANE-BRW	All	**90.60±2.57**	**85.36**

Table 2: Performance on Twitter-Rugby Dataset

Our VANE model outperforms baselines on both tasks. An intuitive explanation is that: comparing with single-view algorithms [5, 7, 12], representations of the VANE framework are more comprehensive across views to support the view adversarial game; and comparing with the multiplex algorithms [8, 14], representations of the VANE framework are more robust by involving the generator. To further verify our guess, we design the following ablation study.

4.4 Ablation Study

In Table 3, we observe that our VANE framework cannot exploit the topology information effectively without the guide of locality constraints in each specific view, and the generator indeed improves performance by improving the robustness of node presentations.

4.5 Stability Analysis

We show the evolution of the loss during training with iteration (i.e. a mini-batch updating) in Figure 3, where we get the loss of each part with other parts fixed. We observe the loss of each part of the model keeps stable during the training process.

Model	Locality Constraints	Node Representation Generator	Accuracy (%)	
			Node Classification	Link Prediction
VANE-BRW	No	No	19.28±4.03	50.03
	No	Yes	17.59±3.33	60.49
	Yes	No	84.70±4.60	81.29
	Yes	Yes	**90.60±2.57**	**85.36**

Table 3: Ablation Study of VANE-BRW on the Twitter-Rugby

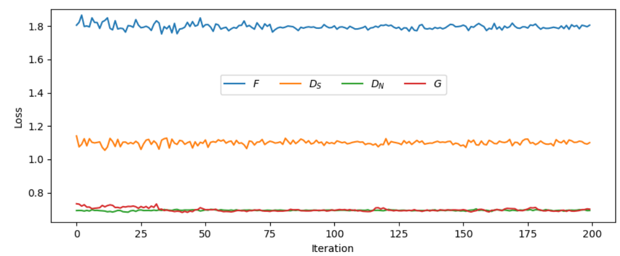

Figure 3: Loss of Different Parts of VANE during Training.

5 CONCLUSION

We propose a view-adversarial multi-view network embedding framework (VANE) for comprehensive and robust node representations across different views via two adversarial games. Extensive experiments show the effectiveness of the our VANE framework.

ACKNOWLEDGEMENT

This work is supported by the National Science Foundation (1947135, 2003924, 1939725, 1947203 and 2002540), the U.S. Department of Homeland Security under Grant Award Number 17STQAC00001-03-03 and Ordering Agreement Number HSHQDC-16-A-B0001. The views and conclusions are those of the authors and should not be interpreted as representing the official policies of the funding agencies or the government.

REFERENCES

[1] Yukuo Cen, Xu Zou, Jianwei Zhang, Hongxia Yang, Jingren Zhou, and Jie Tang. 2019. Representation Learning for Attributed Multiplex Heterogeneous Network. In *KDD 2019*.
[2] Quanyu Dai, Qiang Li, Jian Tang, and Dan Wang. 2018. Adversarial network embedding. In *AAAI 2018*.
[3] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *NeurIPS 2014*.
[4] Derek Greene and Pádraig Cunningham. 2013. Producing a unified graph representation from multiple social network views. In *WedSci 2013*.
[5] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *KDD 2016*.
[6] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* (1997).
[7] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *KDD 2014*.
[8] Meng Qu, Jian Tang, Jingbo Shang, Xiang Ren, Ming Zhang, and Jiawei Han. 2017. An attention-based collaboration framework for multi-view network representation learning. In *CIKM 2017*.
[9] Yiwei Sun, Suhang Wang, Tsung-Yu Hsieh, Xianfeng Tang, and Vasant G. Honavar. 2019. MEGAN: A Generative Adversarial Network for Multi-View Network Embedding. In *IJCAI 2019*.
[10] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding. In *WWW 2015*.
[11] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. 2008. Arnetminer: extraction and mining of academic social networks. In *KDD 2008*.
[12] Hongwei Wang, Jia Wang, Jialin Wang, Miao Zhao, Weinan Zhang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. Graphgan: Graph representation learning with generative adversarial nets. In *AAAI 2018*.
[13] Pei Yang, Qi Tan, Hanghang Tong, and Jingrui He. 2019. In *KDD 2019*.
[14] Hongming Zhang, Liwei Qiu, Lingling Yi, and Yangqiu Song. 2018. Scalable Multiplex Network Embedding. In *IJCAI 2018*.