mvn2vec: Preservation and Collaboration in Multi-View Network Embedding

Yu Shi[†§]   Fangqiu Han[‡] Xinwei He[†] Xinran He[‡] Carl Yang[†] Luo Jie[‡] Jiawei Han[†]

[†]University of Illinois at Urbana-Champaign, Urbana, IL USA   [‡]Snap Inc., Los Angeles, CA USA

[†]{yushi2, xhe17, jiyang3, hanj}@illinois.edu   [‡]{fangqiu.han, xhe2, roger.luo}@snap.com

ABSTRACT

Multi-view networks are ubiquitous in real-world applications. In order to extract knowledge or business value, it is of interest to transform such networks into representations that are easily machine-actionable. Meanwhile, network embedding has emerged as an effective approach to generate distributed network representations. Therefore, we are motivated to study the problem of multi-view network embedding, with a focus on the characteristics that are specific and important in embedding this type of networks. In our practice of embedding real-world multi-view networks, we identify two such characteristics, which we refer to as *preservation* and *collaboration*. We then explore the feasibility of achieving better embedding quality by simultaneously modeling *preservation* and *collaboration*, and propose the mvn2vec algorithms. With experiments on a series of synthetic datasets, an internal Snapchat dataset, and two public datasets, we further confirm the presence and importance of *preservation* and *collaboration*. These experiments also demonstrate that better embedding can be obtained by simultaneously modeling the two characteristics, while not over-complicating the model or requiring additional supervision.

KEYWORDS

Multi-view networks, network embedding, graph mining, representation learning.

1 INTRODUCTION

In real-world applications, objects can be associated with different types of relations. These objects and their relationships can be naturally represented by multi-view networks, which are also known as multiplex networks or multi-view graphs [3, 7–9, 11, 14, 15, 22, 28–30]. As shown in Figure 1a, a multi-view network consists of multiple network views, where each view corresponds to a type of edge, and all views share the same set of nodes. In ecology, a multi-view network can be used to represent the relation among species, where each node stands for a species, and the six views represent predation, competition, symbiosis, parasitism, protocooperation, and commensalism, respectively. On social networking services, a four-view network upon users can be used to describe the widely seen social relationship and interaction including friendship, following, message exchange, and post viewing. With such vast availability of multi-view networks, one could be interested in extracting knowledge or business value from data. In order to achieve this goal with the progressively developed computing power, it is of interest to
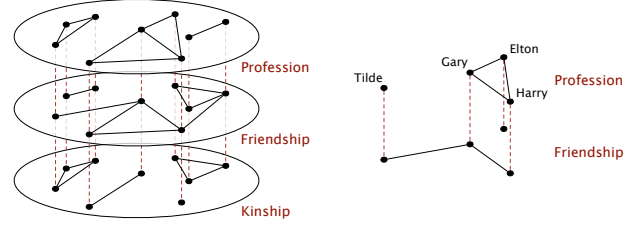
(a) A toy multi-view network.

(b) A close-up look of profession view and friendship view.

Figure 1: A toy example of multi-view networks where each node represents a person and the three views correspond to three types of interpersonal relations. Co-workers are linked in the profession view, friends are linked in the friendship view, and relatives are linked in the kinship view.

first transform the multi-view networks into a different form of representations that are more machine actionable.

Network embedding has emerged as a scalable representation learning method that generates distributed node representations for networked data [6, 16, 23, 25]. Specifically, network embedding projects networks into embedding spaces, where nodes are represented by embedding vectors. With the semantic information of each node encoded, these vectors can be directly used as node features in various downstream applications [6, 16, 23]. Motivated by the success of network embedding in representing homogeneous networks [6, 13, 16, 17, 23, 25], where nodes and edges are untyped, we believe it is important to study the problem of embedding multi-view networks.

To design embedding algorithms for multi-view networks, the major challenge lies in how to make use of the type information on edges from different views. As a result, we are interested in investigating into the following two problems:

(1) With the availability of multiple edge types, what are the characteristics that are *specific* and *important* to multi-view network embedding?
(2) Can we achieve better embedding quality by modeling these characteristics jointly?

To answer the first problem, we identify two characteristics, *preservation* and *collaboration*, from our practice of embedding real-world multi-view networks. We describe the concepts of *preservation* and *collaboration* as follows. **Collaboration** – In some datasets, edges between the same pair of nodes may be observed in different views due to shared latent reasons. For instance, in a social network, if we observe an edge between a user pair in either the message exchange view or the post viewing view, likely these two users are happy to be associated with each other. In such scenario, these

views may complement each other, and embedding them jointly may potentially yield better results than embedding them independently. We call such synergetic effect in jointly embedding multiple views by *collaboration*. The feasibility of enjoying this synergetic effect is also the main intuition behind most existing multi-view network algorithms [7–9, 11, 15, 22, 28–30]. **Preservation** – On the other hand, it is possible for different network views to have different semantic meanings; it is also possible that a portion of nodes have completely disagreeing edges in different views since edges in different views are formed due to distinct latent reasons. For example, professional relationship may not always align well with friendship. If we embed the profession view and the friendship view in Figure 1b into the same embedding space, the embedding of Gary will be close to both Tilde and Elton. As a result, the embedding of Tilde will also not be too distant from Elton due to transitivity. However, this is not a desirable result, because Tilde and Elton are not closely related in terms of either profession or friendship according to the original multi-view network. In other words, embedding in this way fails to preserve the unique information carried by different network views. We refer to such need for preserving unique information carried by different views as *preservation*. The detailed discussion of the presence and importance of *preservation* and *collaboration* is presented in Section 4.

Furthermore, it is also possible for *preservation* and *collaboration* to co-exist in the same multi-view network. Two scenarios can result in this situation: (i) a pair of views are generated from very similar latent reason, while another pair of views carries completely different semantic meanings; and more subtly (ii) for the same pair of views, one portion of nodes have consistent edges in different views, while another portion of nodes have totally disagreeing edges in different views. One example of the latter scenario is that professional relationship does not align well with friendship in some cultures, whereas co-workers often become friends in certain other cultures [1]. Therefore, we are also interested in exploring the feasibility of achieving better embedding quality by modeling *preservation* and *collaboration* simultaneously, and we address this problem in Section 5 and beyond.

We summarize our contributions as follows. (i) We propose to study the characteristics that are specific and important to multi-view network embedding, and identify *preservation* and *collaboration* as two such characteristics from the practice of embedding real-world multi-view networks. (ii) We explore the feasibility of attaining better embedding by simultaneously modeling *preservation* and *collaboration*, and propose two multi-view network embedding methods – MVN2VEC-CON and MVN2VEC-REG. (iii) We conduct experiments with various downstream applications on a series of synthetic datasets and three real-world multi-view networks, including an internal dataset sampled from the Snapchat social network. These experiments corroborate the presence and importance of *preservation* and *collaboration*, and demonstrate the effectiveness of the proposed methods.

2 RELATED WORK

Network embedding has recently emerged as an efficient and effective approach for learning distributed node representations. Instead of leveraging spectral properties of networks as commonly seen in traditional unsupervised feature learning approaches [2, 20, 24, 26], most network embedding methods are designed atop local properties of networks that involve links and proximity among nodes [6, 13, 16, 17, 23, 25]. Such methodology with focus on local properties has been shown to be more scalable. The designs of many recent network embedding algorithms trace to the skip-gram model [12] that aims to learn distributed representation for words in natural language processing, under the assumption that words with similar context should have similar embedding. To fit in the skip-gram model, various strategies have been proposed to define the context of a node in the network scenario [6, 16, 17, 23]. Beyond the skip-gram model, embedding methods for preserving certain other network properties can also be found in the literature [13, 25].

Meanwhile, multi-view networks have been extensively studied as a special type of networks, motivated by their ubiquitous presence in real-world applications. However, most existing methods for multi-view networks aim to bring performance boost in traditional tasks, such as clustering [8, 9, 11, 30], classification [22, 29], and dense subgraph mining [7, 15, 28]. The above methods aim to improve the performance of specific applications, but do not directly study distributed representation learning for multi-view networks. Another line of research on multi-view networks focuses on analyzing interrelations among different views, such as revealing such interrelations via correlation between link existence and network statistics [3, 14]. These works do not directly address how such interrelations can impact the embedding learning of multi-view networks.

Meng et al. [18] recently propose to embed multi-view networks for a given task by linearly combining the embeddings learned from different network views. This work studies a problem different from ours since supervision is required for their framework, while we focus on the unsupervised scenario. Also, their work attends to weighing different views according to their informativeness in a specific task, while we aim at identifying and leveraging the principles when extending a network embedding method from the homogeneous scenario to the multi-view scenario. Moreover, their work does not model *preservation*, one of the characteristics that we deem important for multi-view network embedding, because their final embedding derived via linear combination is a trade-off between representations from all views. Another group of related studies focus on the problem of jointly modeling multiple network views using latent space models [4, 5, 21]. These work again does not model *preservation*.

3 PRELIMINARIES

Definition 3.1 (Multi-View Network). A multi-view network $G = (\mathcal{U}, \{\mathcal{E}^{(v)}\}_{v \in \mathcal{V}})$ is a network consisting of a set $\mathcal{U}$ of nodes and a set $\mathcal{V}$ of views, where $\mathcal{E}^{(v)}$ consists of all edges in view $v \in \mathcal{V}$. If a multi-view network is weighted, then there exists a weight mapping $w : \{\mathcal{E}^{(v)}\}_{v \in \mathcal{V}} \to \mathbb{R}$ such that $w_{uu'}^{(v)} := w(e_{uu'}^{(v)})$ is the weight of the edge $e_{uu'}^{(v)} \in \mathcal{E}^{(v)}$, which joints nodes $u \in \mathcal{U}$ and $u' \in \mathcal{U}$ in view $v \in \mathcal{V}$.

Additionally, when context is clear, we use the network view v of multi-view network $G = (\mathcal{U}, \{\mathcal{E}^{(v)}\}_{v \in \mathcal{V}})$ to denote the untyped network $G^{(v)} = (\mathcal{U}, \mathcal{E}^{(v)})$.

Table 1: Summary of symbols

Symbol	Definition
$\mathcal{V}$	The set of all network views
$\mathcal{U}$	The set of all nodes
$\mathcal{E}^{(v)}$	The set of all edges in view $v \in \mathcal{V}$
$\mathcal{W}^{(v)}$	The list of random walk pairs from view $v \in \mathcal{V}$
$\mathbf{f}_u$	The final embedding of node $u \in \mathcal{U}$
$\mathbf{f}_u^v$	The center embedding of node $u \in \mathcal{U}$ w.r.t. view $v \in \mathcal{V}$
$\tilde{\mathbf{f}}_u^v$	The context embedding of node $u \in \mathcal{U}$ w.r.t. view $v \in \mathcal{V}$
$\theta \in [0, 1]$	The hyperparameter on parameter sharing in MVN2VEC-CON
$\gamma \in \mathbb{R}_{\geq 0}$	The hyperparameter on regularization in MVN2VEC-REG
$D \in \mathbb{N}$	The dimension of the embedding space

Definition 3.2 (Network Embedding). Network embedding aims at learning a (center) embedding $\mathbf{f}_u \in \mathbb{R}^D$ for each node $u \in \mathcal{U}$ in a network, where $D \in \mathbb{N}$ is the dimension of the embedding space.

Besides the center embedding $\mathbf{f}_u \in \mathbb{R}^D$, a family of popular algorithms [12, 23] also deploy a context embedding $\tilde{\mathbf{f}}_u \in \mathbb{R}^D$ for each node u. Moreover, when the learned embedding is used as the feature vector for downstream applications, we take the center embedding of each node as feature following the common practice in algorithms involving context embedding.

4 PRESERVATION AND COLLABORATION IN MULTI-VIEW NETWORK EMBEDDING

In this section, we elaborate on the intuition and presence of *preservation* and *collaboration* – the two characteristics that we have introduced in Section 1 and deem important for multi-view network embedding. In particular, we first describe and investigate the motivating phenomena that are observed in the practice of embedding real-world multi-view networks. Then, we discuss how they can be explained by the two proposed characteristics.

Two straightforward approaches for embedding multi-view networks. Most existing network embedding methods [6, 13, 16, 17, 23, 25] are designed for homogeneous networks, where nodes and edges are untyped, while we are interested in studying the problem of embedding multi-view networks. To extend any untyped network embedding algorithm to multi-view networks, two straightforward yet practical approaches exist. We refer to these two approaches as the *independent* model and the *one-space* model.

Using any untyped network embedding method, we denote $\mathbf{f}_u^v \in \mathbb{R}^{d(v)}$ the (center) embedding of node $u \in \mathcal{U}$ achieved by embedding only the view $v \in \mathcal{V}$ of the multi-view network, where $d(v)$ is the dimension of the embedding space for network view v. With such notation, the *independent* model and the *one-space* model are given as follows.

- **Independent**. Embed each view independently, and then concatenate to derive the final embedding $\mathbf{f}_u$. That is,

$$\mathbf{f}_u = \bigoplus_{v \in \mathcal{V}} \mathbf{f}_u^v \in \mathbb{R}^D, \tag{1}$$

  where $D = \sum_{v \in \mathcal{V}} d(v)$, and $\bigoplus$ represents concatenation. In other words, the embedding of each node in the *independent* model resides in the direct sum of multiple embedding spaces. This approach preserves the information embodied in each view, but do not allow collaboration across different views in the embedding learning process.

Table 2: Embedding quality of two real-world multi-view networks using the *independent* model and the *one-space* model.

Dataset	Metric	*independent*	*one-space*
YouTube	ROC-AUC	**0.931**	0.914
	AUPRC	**0.745**	0.702
Twitter	ROC-AUC	0.724	**0.737**
	AUPRC	0.447	**0.466**

- **One-space**. Let the embedding for different views to share parameters when learning the final embedding $\mathbf{f}_u$. That is,

$$\mathbf{f}_u = \mathbf{f}_u^v \in \mathbb{R}^D, \ \forall v \in \mathcal{V}, \tag{2}$$

  where $D = d(v)$ for all $v \in \mathcal{V}$. In other words, each dimension of the final embedding space correlates with all views of the concerned multi-view network. This approach enables different views to collaborate in learning a unified embedding, but do not preserve information specifically carried by each view. This property of the *one-space* model is corroborated by experiment presented in Section 6.5.

In either of the above two approaches, the same treatment to the center embedding is applied to the context embedding when applicable. It is also worth noting that the embedding learned by the *one-space* model cannot be obtained by linearly combining $\{\mathbf{f}_u^v\}_{v \in \mathcal{V}}$ in the *independent* model. This is because most network embedding models are non-linear models.

Embedding real-word multi-view networks by straightforward approaches. In this paper, *independent* and *one-space* are implemented on top of a random walk plus skip-gram approach as widely seen in the literature [6, 16, 17]. The experiment setup and results are concisely introduced at this point, while detailed description of algorithm, datasets, and more comprehensive experiment results are deferred to Section 5 and 6. Two networks, YouTube and Twitter, are used in these exploratory experiments with users being nodes on each network. YouTube has three views representing common videos (cmn-vid), common subscribers (cmn-sub), and common friends (cmn-fnd) shared by each pair of users, while Twitter has two views corresponding to replying (reply) and mentioning (mention) among users. The downstream evaluation task is to infer whether two users are friends, and the results are presented in Table 2.

It can be seen that the *independent* model consistently outperformed the *one-space* model in the YouTube experiment, while the *one-space* model outperformed the *independent* model in Twitter. These exploratory experiments make it clear that neither of the two straightforward approaches is categorically superior to the other. Furthermore, we interpret the varied performance of the two approaches by the varied extent of needs for modeling *preservation* and modeling *collaboration* when embedding different networks. Specifically, recall that the *independent* model only captures *preservation*, while *one-space* only captures *collaboration*. As a result, we speculate if a certain dataset craves for more *preservation* than *collaboration*, the *independent* model would outperform the *one-space* model, otherwise, the *one-space* model would win.

In order to corroborate our interpretation of the results, we further examine the involved datasets, and look into the agreement between information carried by different network views. We achieve this by a Jaccard coefficient–based measurement, where the
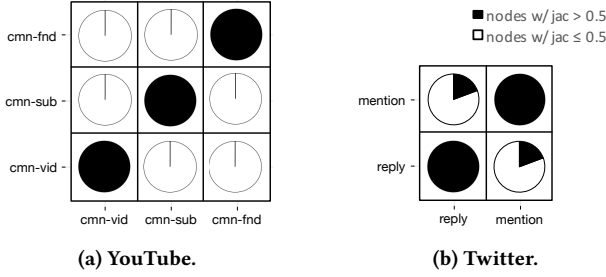
(a) YouTube.

(b) Twitter.

Figure 2: Agreement between information carried by each pair of network views given by a Jaccard coefficient based measurement.

Jaccard coefficient is a similarity measure with range $[0, 1]$, defined as $J(S_1, S_2) = |S_1 \cap S_2|/|S_1 \cup S_2|$ for set S_1 and set S_2. Given a pair of views in a multi-view network, a node can be connected to a different set of neighbors in each of the two network views. The Jaccard coefficient between these two sets of neighbors can then be calculated. In Figure 2, we apply this measurement on the YouTube dataset and the Twitter data, respectively, and illustrate the proportion of nodes with the Jaccard coefficient greater than 0.5 for each pair of views.

As presented in Figure 2, little agreement exists between each pair of different views in YouTube. As a result, it is not surprising that *collaboration* among different views is not as needed as *preservation* in the embedding learning process. On the other hand, a substantial portion of nodes have Jaccard coefficient greater than 0.5 over different views in the Twitter dataset. It is therefore also not surprising to see modeling *collaboration* brings about more benefits than modeling *preservation* in this case.

5 THE MVN2VEC MODELS

In the previous section, *preservation* and *collaboration* are identified as important characteristics for multi-view network embedding. In the extreme cases, where only *preservation* is needed – each view carries a distinct semantic meaning – or only *collaboration* is needed – all views carry the same semantic meaning – it is advisable to choose between *independent* and *one-space* to embed a multi-view network. However, it is of interest to study the also likely scenario where both *preservation* and *collaboration* co-exist in given multi-view networks. Therefore, we are motivated to explore the feasibility of achieving better embedding by simultaneously modeling both characteristics. To this end, we propose and experiment with two approaches that capture both characteristics, without over-complicating the model or requiring additional supervision. These two approaches are named MVN2VEC-CON and MVN2VEC-REG, where MVN2VEC is short for **m**ulti-**v**iew **n**etwork **to vec**tor, while CON and REG stand for constrained and regularized, respectively.

As with the notation convention in Section 4, we denote $\mathbf{f}_u^v \in \mathbb{R}^{d(v)}$ and $\tilde{\mathbf{f}}_u^v \in \mathbb{R}^{d(v)}$ the center and context embedding, respectively, of node $u \in \mathcal{U}$ for view $v \in \mathcal{V}$. Further given the network view v, *i.e.*, $G^{(v)} = (\mathcal{U}, \mathcal{E}^{(v)})$, we use an intra-view loss function to measure how well the current embedding can represent the original network view

$$l(\{\mathbf{f}_u^v, \tilde{\mathbf{f}}_u^v\}_{u \in \mathcal{U}} | G^{(v)}). \qquad (3)$$

We defer the detailed definition of this loss function (Eq. (3)) to a later point of this section. Moreover, we let $d(v) = D/|\mathcal{V}| \in \mathbb{N}$ for

all $v \in \mathcal{V}$ out of convenience for model design. To further incorporate multiple views with the intention to model both *preservation* and *collaboration*, two approaches are proposed as follows.

MVN2VEC-CON. The MVN2VEC-CON model does not enforce further design on the center embedding $\{\mathbf{f}_u^v\}_{u \in \mathcal{U}}$ in the hope of preserving the semantics of each individual view. To reflect *collaboration*, MVN2VEC-CON includes further constraints on the context embedding for parameter sharing across different views

$$\tilde{\mathbf{f}}_u^v = \varphi_\theta^v(\{\tilde{\mathbf{g}}_u^{v'}\}_{v' \in \mathcal{V}}) := (1 - \theta) \cdot \tilde{\mathbf{g}}_u^v + \frac{\theta}{|\mathcal{V}|} \cdot \sum_{v' \in \mathcal{V}} \tilde{\mathbf{g}}_u^{v'}, \quad (4)$$

where $\theta \in [0, 1]$ is a hyperparameter controlling the extend to which model parameters are shared. The greater the value of θ, the more the model enforces parameter sharing and thereby encouraging more *collaboration* across different views. This design aims at allowing different views to collaborate by passing information via the shared parameters in the embedding learning process. That is, the MVN2VEC-CON model solves the following optimization problem

$$\min_{\{\mathbf{f}_u^v, \tilde{\mathbf{g}}_u^v\}_{u \in \mathcal{U}, v \in \mathcal{V}}} \sum_{v \in \mathcal{V}} l(\{\mathbf{f}_u^v, \varphi_\theta^v(\{\tilde{\mathbf{g}}_u^{v'}\}_{v' \in \mathcal{V}})\}_{u \in \mathcal{U}} | G^{(v)}), \quad (5)$$

where $\varphi_\theta^v(\{\tilde{\mathbf{g}}_u^{v'}\}_{v' \in \mathcal{V}})$ is defined in Eq. (4). After model learning, the final embedding for node u is given by $\mathbf{f}_u = \bigoplus_{v \in \mathcal{V}} \mathbf{f}_u^v$. We note that in the extreme case when θ is set to be 0, the model will be identical to the *independent* model discussed in Section 4.

MVN2VEC-REG. In stead of setting hard constraints on how parameters are shared across different views, the MVN2VEC-REG model regularizes the embedding across different views and solves the following optimization problem

$$\min_{\{\mathbf{f}_u^v, \tilde{\mathbf{f}}_u^v\}_{u \in \mathcal{U}, v \in \mathcal{V}}} \sum_{v \in \mathcal{V}} l(\{\mathbf{f}_u^v, \tilde{\mathbf{f}}_u^v\}_{u \in \mathcal{U}} | G^{(v)}) + \gamma \cdot [\mathcal{R}^v + \tilde{\mathcal{R}}^v], \quad (6)$$

where $\|\cdot\|_2$ is the *l*-2 norm, $\mathcal{R}^v = \sum_{u \in \mathcal{U}} \left\| \mathbf{f}_u^v - \frac{1}{|\mathcal{V}|} \sum_{v' \in \mathcal{V}} \mathbf{f}_u^{v'} \right\|_2^2$, $\tilde{\mathcal{R}}^v = \sum_{u \in \mathcal{U}} \left\| \tilde{\mathbf{f}}_u^v - \frac{1}{|\mathcal{V}|} \sum_{v' \in \mathcal{V}} \tilde{\mathbf{f}}_u^{v'} \right\|_2^2$, and $\gamma \in \mathbb{R}_{\geq 0}$ is a hyperparameter. This model captures *preservation* again by letting $\{\mathbf{f}_u^v\}_{u \in \mathcal{U}}$ and $\{\tilde{\mathbf{f}}_u^v\}_{u \in \mathcal{U}}$ to reside in the embedding subspace specific to view $v \in \mathcal{V}$, while each of these subspaces are distorted via cross-view regularization to model *collaboration*. Similar to the MVN2VEC-CON model, the greater the value of the hyperparameter γ, the more the *collaboration* is encouraged, and the model is identical to the *independent* model when $\gamma = 0$.

Intra-view loss function. There are many possible approaches to formulate the intra-view loss function in Eq. (3). In our framework, we adopt the random walk plus skip-gram approach, which is one of the most common methods used in the literature [6, 16, 17]. Specifically, for each view $v \in \mathcal{V}$, multiple rounds of random walks are sampled starting from each node in $G^{(v)} = (\mathcal{U}, \mathcal{E}^{(v)})$. Along any random walk, a node $u \in \mathcal{U}$ and a neighboring node $n \in \mathcal{U}$ constitute one random walk pair, and a list $\mathcal{W}^{(v)}$ of random walk pairs can thereby be derived. We defer the detailed description on the generation of $\mathcal{W}^{(v)}$ to a later point in this section. The

Algorithm 1: MVN2VEC-CON and MVN2VEC-REG

Input :the multi-view network $G = (\mathcal{U}, \{\mathcal{E}^{(v)}\}_{v \in \mathcal{V}})$ and
the hyperparameters
Output:the final embedding $\{\mathbf{f}_u\}_{u \in \mathcal{U}}$
begin
    for $v \in \mathcal{V}$ **do**
        Sample a list $\mathcal{W}^{(v)}$ of random walk pairs
    Join and shuffle the lists of random walk pairs from all
        views to form a new random walk pair list $\mathcal{W}$
    for *each epoch* **do**
        // The following for-loop is parallelized
        for $(u, n) \in \mathcal{W}$ **do**
            if *using model* MVN2VEC-CON **then**
                Update $\{\mathbf{f}_u^v, \tilde{\mathbf{g}}_u^v\}_{u \in \mathcal{U}, v \in \mathcal{V}}$ with one step
                descent using gradients in Eq. (9)–(11)
            if *using model* MVN2VEC-REG **then**
                Update $\{\mathbf{f}_u^v, \tilde{\mathbf{f}}_u^v\}_{u \in \mathcal{U}, v \in \mathcal{V}}$ with one step
                descent using gradients in Eq. (12)–(14)
    for $u \in \mathcal{U}$ **do**
        Derive the embedding for node u by $\mathbf{f}_u = \bigoplus_{v \in \mathcal{V}} \mathbf{f}_u^v$

---

intra-view function is then given by

$$l(\{\mathbf{f}_u^v, \tilde{\mathbf{f}}_u^v\}_{u \in \mathcal{U}} | G^{(v)}) = - \sum_{(u,n) \in \mathcal{W}^{(v)}} \log p^{(v)}(n|u), \quad (7)$$

where

$$p^{(v)}(n|u) = \frac{\exp\left(\mathbf{f}_u^v \cdot \tilde{\mathbf{f}}_n^v\right)}{\sum_{n' \in \mathcal{U}} \exp\left(\mathbf{f}_u^v \cdot \tilde{\mathbf{f}}_{n'}^v\right)}. \quad (8)$$

Model inference. To optimize the objectives in Eq. (5) and (6), we opt to asynchronous stochastic gradient descent (ASGD) [19] following existing skip-gram–based algorithms [6, 12, 16, 17, 23]. In this regard, $\mathcal{W}^{(v)}$ from all views are joined and shuffled to form a new list $\mathcal{W}$ of random walk pairs for all views. Then each step of ASGD draws one random walk pair from $\mathcal{W}$, and updates corresponding model parameters with one-step gradient descent.

Moreover, due to the existence of partition function in Eq. (8), computing gradients of Eq. (5) and (6) is unaffordable with Eq. (7) being their parts. Negative sampling is hence adopted as in other skip-gram–based methods [6, 12, 16, 17, 23], which approximates $\log p^{(v)}(n|u)$ in Eq. (7) by

$$- \log \sigma(\mathbf{f}_u^v \cdot \tilde{\mathbf{f}}_n^v) - \sum_{i=1}^{K} \mathbb{E}_{n_i' \sim P^{(v)}} \log \sigma(-\mathbf{f}_u^v \cdot \tilde{\mathbf{f}}_{n_i'}^v),$$

where $\sigma(x) = 1/(1 + \exp(-x))$ is the sigmoid function, K is the negative sampling rate, $P^{(v)}(u) \propto \left[D_u^{(v)}\right]^{3/4}$ is the noise distribution, and $D_u^{(v)}$ is the number of occurrences of node u in $\mathcal{W}^{(v)}$ [12].

With negative sampling, the objective function involving one walk pair (u, n) drawn from view v in MVN2VEC-CON is

$$O_{\text{CON}} = \log \sigma(\mathbf{f}_u^v \cdot \varphi_\theta^v(\{\tilde{\mathbf{g}}_n^{v'}\}_{v' \in \mathcal{V}}))$$
$$+ \sum_{i=1}^{K} \mathbb{E}_{n_i' \sim P^{(v)}} \log \sigma(-\mathbf{f}_u^v \cdot \varphi_\theta^v(\{\tilde{\mathbf{g}}_{n_i'}^{v'}\}_{v' \in \mathcal{V}})).$$

On the other hand, the objective function involving (u, n) from view v in MVN2VEC-REG is

$$O_{\text{REG}} = \log \sigma(\mathbf{f}_u^v \cdot \tilde{\mathbf{f}}_n^v) + \gamma \cdot (\mathcal{R}_u^v + \tilde{\mathcal{R}}_n^v)$$
$$+ \sum_{i=1}^{K} \left[\mathbb{E}_{n_i' \sim P^{(v)}} \log \sigma(-\mathbf{f}_u^v \cdot \tilde{\mathbf{f}}_{n_i'}^v) + \gamma \cdot (\mathcal{R}_u^v + \tilde{\mathcal{R}}_{n_i'}^v) \right],$$

and $\mathcal{R}_u^v = \left\| \mathbf{f}_u^v - \frac{1}{|\mathcal{V}|} \sum_{v' \in \mathcal{V}} \mathbf{f}_u^{v'} \right\|_2^2$, $\tilde{\mathcal{R}}_u^v = \left\| \tilde{\mathbf{f}}_u^v - \frac{1}{|\mathcal{V}|} \sum_{v' \in \mathcal{V}} \tilde{\mathbf{f}}_u^{v'} \right\|_2^2$. The gradients of the above two objective function used for ASGD are provided in the appendix.

Random walk pair generation. Without additional supervision, we assume equal importance of different network views in learning embedding, and sample the same number $N \in \mathbb{N}$ of random walks from each view. To determine this number, we denote $n^{(v)}$ the number of nodes that are not isolated from the rest of the network in view $v \in \mathcal{V}$, $n_{max} := \max\{n^{(v)} : v \in \mathcal{V}\}$, and let $N := M \cdot n_{max}$, where M is a hyperparameter to be specified.

Given a network view $v \in \mathcal{V}$, we generate random walk pairs as in existing work [6, 16, 17]. Specifically, each random walk is of length $L \in \mathbb{N}$, and $\lfloor N/n^{(v)} \rfloor$ or $\lceil N/n^{(v)} \rceil$ random walks are sampled from each non-isolated node in view v, yielding a total of N random walks. For each node along any random walk, this node and any other node within a window of size $B \in \mathbb{N}$ constitute a random walk pair that is then added to $\mathcal{W}^{(v)}$.

Finally, we summarize both the MVN2VEC-CON algorithm and the MVN2VEC-REG algorithm in Algorithm 1.

6 EXPERIMENTS

In this section, we further corroborate the intuition of *preservation* and *collaboration*, and demonstrate the feasibility of simultaneously model these two characteristics. We first perform a case study on a series of synthetic multi-view networks that have varied extent of *preservation* and *collaboration*. Next, we introduce the real-world datasets, baselines, and experiment setting for more comprehensive quantitative evaluations. Lastly, we analyze the evaluation results and provide further discussion.

6.1 Case Study – Varied *preservation* and *collaboration* on Synthetic Data

In order to directly study the relative performance of different models on networks with varied extent of *preservation* and *collaboration*, we design a series of synthetic multi-view networks and experiment on a multi-class classification task.

We denote each of these synthetic networks by $S(p)$, where $p \in [0, 0.5]$ is referred to as intrusion probability. Each $S(p)$ has 4,000 nodes and 2 views – v_1 and v_2. Furthermore, each node is associated to one of the 4 class labels – A, B, C, or D – and each class has exactly 1,000 nodes. We first describe the process for generating $S(0)$ before introducing the more general $S(p)$ as follows:

(1) Generate one random network over all nodes with label A or B, and another over all nodes with label C or D. Put all edges in these two random networks into view v_1.
(2) Generate one random network over all nodes with label A or C, and another over all nodes with label B or D. Put all edges in these two random networks into view v_2.
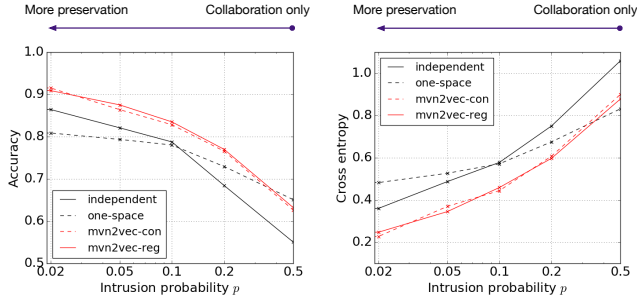
Figure 3: Classification results under accuracy and cross entropy on synthetic networks $S(p)$ with varied intrusion probability p, corresponding to different extent of *preservation* and *collaboration*.

To generate each of the four aforementioned random networks, we adopt the preferential attachment process with $2,000$ nodes and 1 edge to attach from a new node to existing nodes, where the preferential attachment process is a widely used method for generating networks with power-law degree distribution.

With this design for $S(0)$, view v_1 carries the information that nodes labeled A or B should be classified differently from nodes labeled C or D, while v_2 reflects that nodes labeled A or C are different from nodes labeled B or D. More generally, $S(p)$ are generated with the following tweak from $S(0)$: when putting an edge into one of the two views, with probability p, the edge is put into the other view instead of the view specified in the $S(0)$ generation process.

It is worth noting that larger p favors more *collaboration*, while smaller p favors more *preservation*. In the extreme case where $p = 0.5$, only *collaboration* is needed in the network embedding process. This is because every edge has equal probability to fall into view v_1 or view v_2 of $S(0.5)$, and there is hence no information carried specifically by either view that should be preserved.

On each $S(p)$, *independent*, *one-space*, MVN2VEC-CON, and MVN2VEC-REG are tested. On top of the embedding learned by each model, we apply logistic regression with cross entropy to carry out the multi-class evaluation tasks. All model parameters are tuned to the best for each model on a validation dataset sampled from the $4,000$ class labels. Classification accuracy and cross-entropy on a different test dataset are reported in Figure 3.

From Figure 3, we make three observations. (i) *independent* performs better than *one-space* in case p is small – when *preservation* is the dominating characteristic in the network – and *one-space* performs better than *independent* in case p is large – when *collaboration* is dominating. (ii) The two proposed MVN2VEC models perform better than both *independent* and *one-space* except when p is close to 0.5, which implies it is indeed feasible for MVN2VEC to achieve better performance by simultaneously model the two characteristics *preservation* and *collaboration*. (iii) When p is close to 0.5, *one-space* performs the best. This is expected because no *preservation* is needed in $S(0.5)$, and any attempts to additionally model *preservation* shall not boost, if not impair, the performance.

6.2 Data Description and Evaluation Tasks

We perform quantitative evaluations on three real-world multi-view networks: Snapchat, YouTube, and Twitter. The key statistics are summarized in Table 3, and we describe these datasets as follows.

Table 3: Basic statistics for the three real-world multi-view networks, where the number of edges specifies the total edge number from all network views.

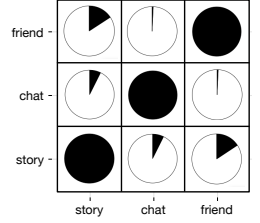| Dataset	# views	# nodes	# edges
Snapchat	3	7,406,859	131,729,903
YouTube	3	14,900	7,977,881
Twitter	2	116,408	183,341

Figure 4: Agreement between each pair of views for Snapchat.

Snapchat. Snapchat is a multimedia social networking service. On the Snapchat multi-view social network, each node is a user, and the three views correspond to friendship, chatting, and story viewing[*]. We perform experiments on the sub-network consisting of all users from Los Angeles. The data used to construct the network are collected from two consecutive weeks in the Spring of 2017. Additional data for downstream evaluation tasks are collected from the following week – henceforth referred to as week 3. We perform a multi-label classification task and a link prediction task on top of the user embedding learned from each network. For classification, we classify whether or not a user views each of the 10 most popular discover channels[†] according to the user viewing history in week 3. For each channel, the users who view this channel are labeled positive, and we randomly select 5 times as many users who do not view this channel as negative examples. These records are then randomly split into training, validation, and test sets. This is a multi-label classification problem that aims at inferring users' preference on different discover channels and can therefore guide product design in content serving. For link prediction, we predict whether two users would view the stories posted by each other in week 3. Negative examples are the users who are friends, but do not have story viewing in the same week. It is worth noting that this definition yields more positive examples than negative examples, which is the cause of a relatively high AUPRC score observed in experiments. These records are then randomly split into training, validation, and test sets with the constraint that a user appears as the viewer of a record in at most one of the three sets. This task aims to estimate the likelihood of story viewing between friends, so that the application can rank stories accordingly.

We also provide the Jaccard coefficient–based measurement on Snapchat in Figure 4. It can be seen that the cross-view agreement between each pair of views in the Snapchat network falls in between YouTube and Twitter presented in Section 2.

YouTube. YouTube is a video-sharing website. We use a dataset made publicly available by the Social Computing Data Repository [27][‡]. From this dataset, a network with three views is constructed, where each node is a core user and the edges in the three views represent the number of common friends, the number of common subscribers, and the number of common favorite videos, respectively. Note that the core users are those from which the author of the dataset crawled the data, and their friends can fall

---

[*]https://support.snapchat.com/en-US/a/view-stories

[†]https://support.snapchat.com/en-US/a/discover-how-to

[‡]http://socialcomputing.asu.edu/datasets/YouTube

out of the scope of the set of core users. Without user label available for classification, we perform only link prediction task on top of the user embedding. This task aims at inferring whether two core users are friends, which has also been used for evaluation by existing research [18]. Each core user forms positive pairs with his or her core friends, and we randomly select 5 times as many non-friend core users to form negative examples. Records are split into training, validation, and test sets as in the link prediction task on Snapchat.

Twitter. Twitter is an online news and social networking service. We use a dataset made publicly available by the Social Computing Data Repository [10]§. From this dataset, a network with two views is constructed, where each node is a user and the edges in the two views represent the number of replies and the number of mentions, respectively. Again, we evaluate by a link prediction task that infers whether two users are friends as in existing research [18]. The same negative example generation method and training–validation–test split method are used as in the YouTube dataset.

For each evaluation task on all three networks, training, validation, and test sets are derived in a shuffle split manner with a 80%–10%–10% ratio. The shuffle split is conducted for 20 times, so that mean and its standard error under each metric can be calculated. Furthermore, a node is excluded from evaluation if it is isolated from other nodes in at least one of the multiple views.

6.3 Baselines and Experimental Setup

In this section, we describe the baselines used to validate the utility of modeling *preservation* and *collaboration*, and the experimental setup for both embedding learning and downstream evaluation tasks.

Baselines. Quantitative evaluation results are obtained by applying downstream learner upon embedding learned by a given embedding method. Therefore, for fair comparisons, we use the same downstream learner in the same evaluation task. Moreover, since our study aims at understanding the characteristics of multi-view network embedding, we build all compared embedding methods from the same random work plus skip-gram approach with the same model inference method, as discussed in Section 5. Specifically, we describe the baseline embedding methods as follows:

- **Independent**. As briefly discussed in Section 4, the *independent* model first embeds each network view independently, and then concatenate them to find the final embedding $\mathbf{f}_u = \bigoplus_{v \in \mathcal{V}} \mathbf{f}_u^v \in \mathbb{R}^D$. This method is equivalent to MVN2VEC-CON when $\theta = 0$, and to MVN2VEC-REG when $\gamma = 0$. It preserves the information embodied in each view, but do not allow collaboration across different views in the embedding process.
- **One-space**. Also discussed in Section 4, the *one-space* model assumes the embedding of the same node to share model parameters across different views $\mathbf{f}_u = \mathbf{f}_u^v \in \mathbb{R}^D$, $\forall v \in \mathcal{V}$. It uses the same strategy to combine random walks generated from different views as with the proposed MVN2VEC methods. *one-space* enables different views to collaborate in learning a unified embedding, but do not preserve information specifically carried by each view.

- **View-merging**. The *view-merging* model first merges all network views into one unified view, and then learn the embedding of this single unified view. In order to comply with the assumed equal importance of different network views, we scale the weights of edges proportionally in each view, so that the total edge weights from all views are the same in the merged network. This method serves as an alternate approach to *one-space* in modeling *collaboration*. The difference between *view-merging* and *one-space* essentially lies in whether or not random walks can cross different views. We note that just like *one-space*, *view-merging* does not model *preservation*.
- **Single-view**. For each network view, the *single-view* model learns embedding from only this view, and neglects all other views. This baseline is used to verify whether introducing more than one view does bring in informative signals in each evaluation task.

Downstream learners. For fair comparisons, we apply the same downstream learner onto the features derived from each embedding method. Specifically, we use the scikit-learn¶ implementation of logistic regression with l-2 regularization and the SAG solver for both classification and link prediction tasks. For each task and each embedding method, we tune the regularization coefficient in the logistic regression to the best on the validation set. Following existing research [23], each embedding vector is normalized onto the unit l-2 sphere before feeding into downstream learners. In multi-label classification tasks, the features fed into the downstream learner is simply the embedding of each node, and we train an independent logistic regression model for each label. In link prediction tasks, features of node pairs are needed, and we derive such features by the Hadamard product of the two involved node embedding vectors as suggested by previous work [6].

Hyperparamters. For *independent*, MVN2VEC-CON, and MVN2VEC-REG, we set embedding space dimension $D = 128 \cdot |\mathcal{V}|$. For *one-space* and *view-merging*, we experiment with both $D = 128$ and $D = 128 \cdot |\mathcal{V}|$, and always report the better result between the two settings. For *single-view*, we set $D = 128$. To generate random walk pairs, we always set $L = 20$ and $B = 3$. For the Snapchat-LA network, we set $M = 10$ due to its large scale, and set $M = 50$ for all other datasets. The negative sampling rate K is set to be 5 for all models, and each model is trained for 1 epoch. In Figure 3, Table 4, Table 5, and Figure 6, θ and γ in the MVN2VEC models are also tuned to the best on the validation dataset. The impact of θ and γ on model performance is further presented and discussed in Section 6.5.

Metrics. For link prediction tasks, we use two widely used metrics: the area under the receiver operating characteristic curve (ROC-AUC) and the area under the precision-recall curve (AUPRC). The receiver operating characteristic curve (ROC) is derived from plotting true positive rate against false positive rate as the threshold varies, and the precision-recall curve (PRC) is created by plotting precision against recall as the threshold varies. Higher values are preferable for both metrics. For multi-label classification tasks, we

---

§https://snap.stanford.edu/data/higgs-twitter.html

¶http://scikit-learn.org/stable/

Table 4: Mean of link prediction results with standard error in brackets.

Dataset	Metric	single-view		independent	one-space	view-merging	MVN2VEC-CON	MVN2VEC-REG
		(worst view)	(best view)					
Snapchat	ROC-AUC	0.587 (0.001)	0.592 (0.001)	0.617 (0.001)	0.603 (0.001)	0.611 (0.001)	0.626 (0.001)	**0.638** (0.001)
	AUPRC	0.675 (0.001)	0.677 (0.002)	0.700 (0.001)	0.688 (0.002)	0.693 (0.002)	0.709 (0.001)	**0.712** (0.002)
YouTube	ROC-AUC	0.831 (0.002)	0.904 (0.002)	**0.931** (0.001)	0.914 (0.001)	0.912 (0.001)	**0.932** (0.001)	**0.934** (0.001)
	AUPRC	0.515 (0.004)	0.678 (0.004)	**0.745** (0.003)	0.702 (0.004)	0.699 (0.004)	**0.746** (0.003)	**0.754** (0.003)
Twitter	ROC-AUC	0.597 (0.001)	0.715 (0.001)	0.724 (0.001)	0.737 (0.001)	0.741 (0.001)	0.727 (0.000)	**0.754** (0.001)
	AUPRC	0.296 (0.001)	0.428 (0.001)	0.447 (0.001)	0.466 (0.001)	0.469 (0.001)	0.453 (0.001)	**0.478** (0.001)

Table 5: Mean of classification results with standard error in brackets on the Snapchat multi-view network.

Dataset	Metric	single-view		independent	one-space	view-merging	MVN2VEC-CON	MVN2VEC-REG
		(worst view)	(best view)					
Snapchat	ROC-AUC	0.634 (0.001)	0.667 (0.002)	0.687 (0.001)	0.675 (0.001)	0.672 (0.001)	**0.693** (0.001)	0.690 (0.001)
	AUPRC	0.252 (0.001)	0.274 (0.002)	0.293 (0.002)	0.278 (0.001)	0.279 (0.001)	**0.298** (0.001)	0.296 (0.002)

also compute the ROC-AUC and the AUPRC for each label, and report the mean value averaged across all labels.

6.4 Quantitative Evaluation Results on Real-World Datasets

The link prediction experiment results on three networks are presented in Table 4. For each dataset, all methods leveraging multiple views outperformed those using only one view, which justifies the necessity of using multi-view networks. Moreover, *one-space* and *view-merging* had comparable performance on each dataset. This is an expected outcome because they both only model *collaboration* and differ from each other merely in whether random walks are performed across network views.

On YouTube, the proposed MVN2VEC models perform as good but do not significantly exceed the baseline *independent* model. Recall that the need for *preservation* in the YouTube network is overwhelmingly dominating as discussed in Section 4. As a result, it is not surprising to see that additionally modeling *collaboration* does not bring about significant performance boost in such extreme case. On Twitter, *collaboration* plays a more important role than *preservation*, as confirmed by the better performance of *one-space* than *independent*. Furthermore, MVN2VEC-REG achieved better performance than all baselines, while MVN2VEC-CON outperformed *independent* by further modeling *collaboration*, but failed to exceed *one-space*. This phenomenon can be explained by the fact that $\{\mathbf{f}_u^v\}_{v \in \mathcal{V}}$ in MVN2VEC-CON are set to be independent regardless of its hyperparameter $\theta \in [0, 1]$, and MVN2VEC-CON's capability of modeling *collaboration* is bounded by this design.

The Snapchat network used in our experiments lies in between YouTube and Twitter in terms of the need for *preservation* and *collaboration*. The proposed two MVN2VEC models both outperformed all baselines under all metrics. In other words, this experiment result shows the feasibility of gaining performance boost by simultaneously model *preservation* and *collaboration* without overcomplicating the model or adding supervision.

The multi-label classification results on Snapchat are presented in Table 4. As with the previous link prediction results, the two MVN2VEC model both outperformed all baselines under all metrics, with a difference that MVN2VEC-CON performed better in this

classification task, while MVN2VEC-REG outperformed better in the previous link prediction task. Overall, while MVN2VEC-CON and MVN2VEC-REG may have different advantages in different tasks, they both outperformed all baselines by simultaneously modeling *preservation* and *collaboration* on the Snapchat network, where both *preservation* and *collaboration* co-exist.

6.5 Hyperparameter Study

Impact of θ for MVN2VEC-CON and γ for MVN2VEC-REG. With results presented in Figure 5, we first focus on the Snapchat network. Starting from $\gamma = 0$, where only *preservation* was modeled, MVN2VEC-REG performed progressively better as more *collaboration* kicked in by increasing γ. The peak performance was reached between 0.01 and 0.1. On the other hand, the performance of MVN2VEC-CON improved as θ grew. Recall that even in case $\theta = 1$, MVN2VEC-CON still have $\{\mathbf{f}_u^v\}_{v \in \mathcal{V}}$ independent in each view. This prevented MVN2VEC-CON from promoting more *collaboration*.

On YouTube, the MVN2VEC models did not significantly outperform *independent* no matter how θ and γ varied due to the dominant need for *preservation* as discussed in Section 4 and 6.4.

On Twitter, MVN2VEC-REG outperformed *one-space* when γ was large, while MVN2VEC-CON could not beat *one-space* for reason discussed in Section 6.4. This also echoed MVN2VEC-CON's performance on Snapchat as discussed in the first paragraph of this section.

Impact of embedding dimension. To rule out the possibility that *one-space* could actually preserve the view-specific information as long as the embedding dimension were set to be large enough, we further carry out the multi-class classification task on $S(0)$ under varied embedding dimensions. Note that $S(0)$ is used in this experiment because it has the need for modeling *preservation* as discussed in Section 6.1. As presented in Figure 6, *one-space* achieves its best performance at $D = 256$, which is worse than *independent* at $D = 256$, let alone the best performance of *independent* at $D = 512$. Therefore, one cannot expect *one-space* to preserve the information carried by different views by employing embedding space with large enough dimension.

Besides, all four models achieve their best performance with D in the vicinity of 256~512. Particularly, *one-space* requires the smallest embedding dimension to reach peak performance. This

(a) Varying θ of MVN2VEC-CON in Snapchat link prediction.

(b) Varying γ of MVN2VEC-REG in Snapchat link prediction.

(c) Varying θ of MVN2VEC-CON in Snapchat classification.

(d) Varying γ of MVN2VEC-REG in Snapchat classification.

(e) Varying θ of MVN2VEC-CON in YouTube link prediction.

(f) Varying γ of MVN2VEC-REG in YouTube link prediction.

(g) Varying θ of MVN2VEC-CON in Twitter link prediction.

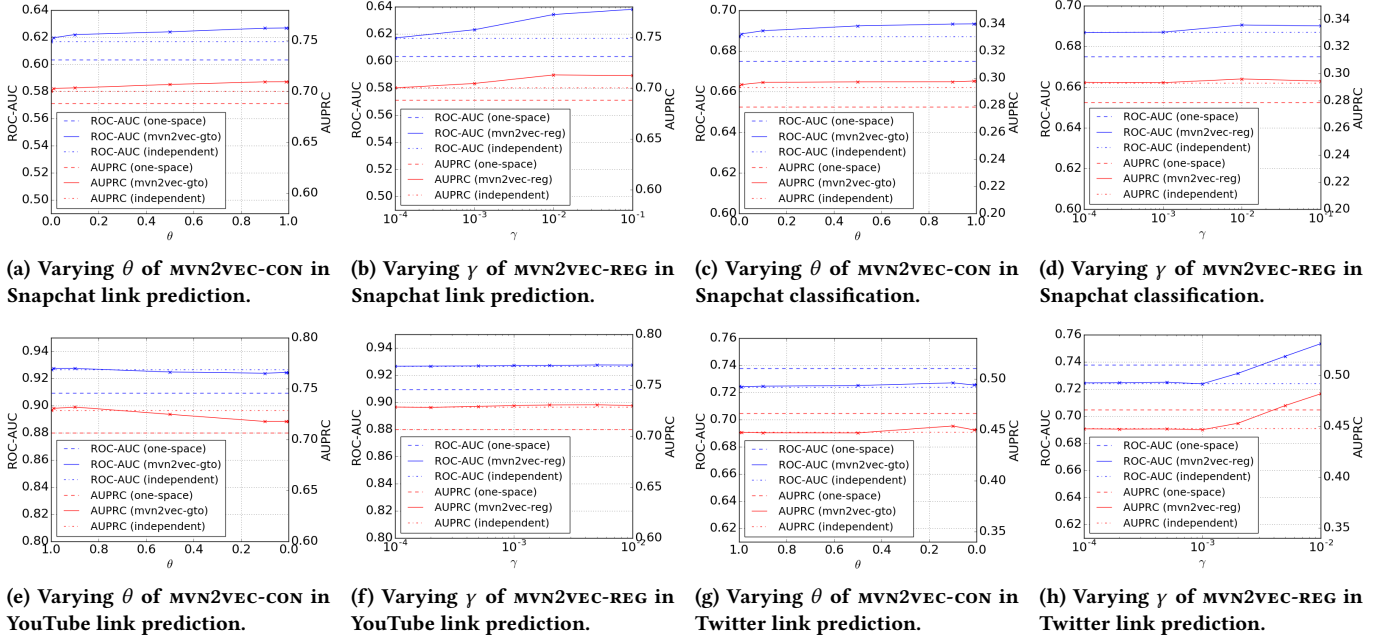(h) Varying γ of MVN2VEC-REG in Twitter link prediction.

Figure 5: Performance of the MVN2VEC models under varying hyperparameters regarding *preservation* and *collaboration*.
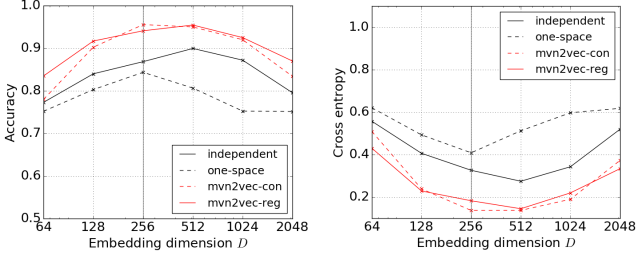

Figure 6: Classification results under accuracy and cross entropy on network $S(0)$ with varied embedding dimension D.

is expected because, unlike the other models, *one-space* does not segment its embedding space to suit multiple views, and hence has more freedom in exploiting an embedding space with given dimension.

7 CONCLUSION AND FUTURE WORK

We studied the characteristics that are specific and important to multi-view network embedding. *preservation* and *collaboration* were identified as two such characteristics in our practice of embedding real-world multi-view networks. We then explored the feasibility of achieving better embedding results by simultaneously modeling *preservation* and *collaboration*, and proposed two multi-view network embedding methods to achieve this objective. Experiments with various downstream evaluation tasks were conducted on a series of synthetic networks and three real-world multi-view networks with distinct sources, including two public datasets and an internal Snapchat dataset. Experiment results corroborated the presence and importance of *preservation* and *collaboration*, and demonstrated the effectiveness of the proposed methods.

Knowing the existence of the identified characteristics, future work includes modeling different extent of *preservation* and *collaboration* for different pairs of views in multi-view embedding. It is also rewarding to explore supervised methods for task-specific multi-view network embedding that are capable of modeling *preservation* and *collaboration* jointly.

APPENDIX

We provide the gradients used for ASGD in the proposed algorithms.

MVN2VEC-CON:

$$\frac{\partial O_{\text{CON}}}{\partial \mathbf{f}_u^v} = \left(1 - \sigma(\mathbf{f}_u^v \cdot \varphi_\theta^v(\{\tilde{\mathbf{g}}_n^{v'}\}_{v' \in \mathcal{V}}))\right) \cdot \varphi_\theta^v(\{\tilde{\mathbf{g}}_n^{v'}\}_{v' \in \mathcal{V}})$$
$$- \sum_{i=1}^{K} \sigma(\mathbf{f}_u^v \cdot \varphi_\theta^v(\{\tilde{\mathbf{g}}_{n_i'}^{v'}\}_{v' \in \mathcal{V}})) \cdot \varphi_\theta^v(\{\tilde{\mathbf{g}}_{n_i'}^{v'}\}_{v' \in \mathcal{V}}), \quad (9)$$

$$\frac{\partial O_{\text{CON}}}{\partial \tilde{\mathbf{g}}_n^{\hat{v}}} = \left(1 - \sigma(\mathbf{f}_u^v \cdot \varphi_\theta^v(\{\tilde{\mathbf{g}}_n^{v'}\}_{v' \in \mathcal{V}}))\right) \cdot \mathbf{f}_u^v \cdot \begin{cases} \theta + \frac{1-\theta}{|\mathcal{V}|}, & \hat{v} = v, \\ \theta, & \hat{v} \neq v, \end{cases} \quad (10)$$

$$\frac{\partial O_{\text{CON}}}{\partial \tilde{\mathbf{g}}_{n_i'}^{\hat{v}}} = -\sigma(\mathbf{f}_u^v \cdot \varphi_\theta^v(\{\tilde{\mathbf{g}}_{n_i'}^{v'}\}_{v' \in \mathcal{V}})) \cdot \mathbf{f}_u^v \cdot \begin{cases} \theta + \frac{1-\theta}{|\mathcal{V}|}, & \hat{v} = v, \\ \theta, & \hat{v} \neq v. \end{cases} \quad (11)$$

MVN2VEC-REG:

$$\frac{\partial O_{\text{REG}}}{\partial \mathbf{f}_u^v} = \left(1 - \sigma(\mathbf{f}_u^v \cdot \tilde{\mathbf{f}}_n^v)\right) \cdot \tilde{\mathbf{f}}_n^v - \sum_{i=1}^{K} \sigma(\mathbf{f}_u^v \cdot \tilde{\mathbf{f}}_{n_i'}^v) \cdot \tilde{\mathbf{f}}_{n_i'}^v$$
$$+ 2\gamma \left(K + 1\right)\left(1 - \frac{1}{|\mathcal{V}|}\right) \cdot \left(\mathbf{f}_u^v - \frac{1}{|\mathcal{V}|} \sum_{v' \in \mathcal{V}} \mathbf{f}_u^{v'}\right), \quad (12)$$

$$\frac{\partial O_{\text{REG}}}{\partial \tilde{\mathbf{f}}_n^v} = \left(1 - \sigma(\mathbf{f}_u^v \cdot \tilde{\mathbf{f}}_n^v)\right) \cdot \mathbf{f}_u^v + 2\gamma\left(1 - \frac{1}{|\mathcal{V}|}\right) \cdot \left(\tilde{\mathbf{f}}_n^v - \frac{1}{|\mathcal{V}|} \sum_{v' \in \mathcal{V}} \tilde{\mathbf{f}}_n^{v'}\right), \quad (13)$$

$$\frac{\partial O_{\text{REG}}}{\partial \tilde{\mathbf{f}}_{n_i'}^v} = -\sigma(\mathbf{f}_u^v \cdot \tilde{\mathbf{f}}_{n_i'}^v) \cdot \mathbf{f}_u^v + 2\gamma\left(1 - \frac{1}{|\mathcal{V}|}\right) \cdot \left(\tilde{\mathbf{f}}_{n_i'}^v - \frac{1}{|\mathcal{V}|} \sum_{v' \in \mathcal{V}} \tilde{\mathbf{f}}_{n_i'}^{v'}\right). \quad (14)$$

Note that in implementation, $|V|$ should be the number of views in which u is associated with at least one edge.

REFERENCES

[1] Jon P Alston. 1989. Wa, guanxi, and inhwa: Managerial principles in Japan, China, and Korea. *Business Horizons* 32, 2 (1989), 26–31.

[2] Mikhail Belkin and Partha Niyogi. 2001. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *NIPS*, Vol. 14. 585–591.

[3] Ove Frank and Krzysztof Nowicki. 1993. Exploratory statistical anlaysis of networks. *Annals of Discrete Mathematics* 55 (1993), 349–365.

[4] Isabella Gollini and Thomas Brendan Murphy. 2014. Joint Modelling of Multiple Network Views. *Journal of Computational and Graphical Statistics* (2014), 00–00.

[5] Derek Greene and Pádraig Cunningham. 2013. Producing a unified graph representation from multiple social network views. In *Web Science Conference*. ACM, 118–121.

[6] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 855–864.

[7] Haiyan Hu, Xifeng Yan, Yu Huang, Jiawei Han, and Xianghong Jasmine Zhou. 2005. Mining coherent dense subgraphs across massive biological networks for functional discovery. *Bioinformatics* 21, suppl 1 (2005), i213–i221.

[8] Abhishek Kumar and Hal Daumé. 2011. A co-training approach for multi-view spectral clustering. In *ICML*. 393–400.

[9] Abhishek Kumar, Piyush Rai, and Hal Daume. 2011. Co-regularized multi-view spectral clustering. In *NIPS*. 1413–1421.

[10] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data. (June 2014).

[11] Jialu Liu, Chi Wang, Jing Gao, and Jiawei Han. 2013. Multi-view clustering via joint nonnegative matrix factorization. In *SDM*, Vol. 13. SIAM, 252–260.

[12] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[13] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. 2016. Asymmetric Transitivity Preserving Graph Embedding.. In *KDD*. 1105–1114.

[14] Philippa Pattison and Stanley Wasserman. 1999. Logit models and logistic regressions for social networks: II. Multivariate relations. *Brit. J. Math. Statist. Psych.* 52, 2 (1999), 169–194.

[15] Jian Pei, Daxin Jiang, and Aidong Zhang. 2005. On mining cross-graph quasi-cliques. In *KDD*. ACM, 228–238.

[16] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 701–710.

[17] Bryan Perozzi, Vivek Kulkarni, Haochen Chen, and Steven Skiena. 2017. Don't Walk, Skip! Online Learning of Multi-scale Network Embeddings. In *Advances in Social Networks Analysis and Mining (ASONAM), 2017 IEEE/ACM International Conference on.*

[18] 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*. ACM.

[19] Benjamin Recht, Christopher Re, Stephen Wright, and Feng Niu. 2011. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in Neural Information Processing Systems*. 693–701.

[20] Sam T Roweis and Lawrence K Saul. 2000. Nonlinear dimensionality reduction by locally linear embedding. *Science* 290, 5500 (2000), 2323–2326.

[21] Michael Salter-Townshend and Tyler H McCormick. 2013. *Latent Space Models for Multiview Network Data*. Technical Report 622. Department of Statistics, University of Washington.

[22] Vikas Sindhwani and Partha Niyogi. 2005. A co-regularized approach to semi-supervised learning with multiple views. In *ICML Workshop on Learning with Multiple Views*.

[23] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1067–1077.

[24] Joshua B Tenenbaum, Vin De Silva, and John C Langford. 2000. A global geometric framework for nonlinear dimensionality reduction. *Science* 290, 5500 (2000), 2319–2323.

[25] Daixin Wang, Peng Cui, and Wenwu Zhu. 2016. Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1225–1234.

[26] Shuicheng Yan, Dong Xu, Benyu Zhang, Hong-Jiang Zhang, Qiang Yang, and Stephen Lin. 2007. Graph embedding and extensions: A general framework for dimensionality reduction. *IEEE transactions on pattern analysis and machine intelligence* 29, 1 (2007).

[27] R. Zafarani and H. Liu. 2009. Social Computing Data Repository at ASU. (2009). http://socialcomputing.asu.edu

[28] Zhiping Zeng, Jianyong Wang, Lizhu Zhou, and George Karypis. 2006. Coherent closed quasi-clique discovery from large dense graph databases. In *KDD*. ACM, 797–802.

[29] Dan Zhang, Fei Wang, Changshui Zhang, and Tao Li. 2008. Multi-View Local Learning.. In *AAAI*. 752–757.

[30] Dengyong Zhou and Christopher JC Burges. 2007. Spectral clustering and transductive learning with multiple views. In *ICML*. ACM, 1159–1166.