Unsupervised Differentiable Multi-aspect Network Embedding

Chanyoung Park[1], Carl Yang[1], Qi Zhu[1], Donghyun Kim[3], Hwanjo Yu[2*], Jiawei Han[1]

[1]Department of Computer Science, University of Illinois at Urbana-Champaign, IL, USA

[2]Pohang University of Science and Technology (POSTECH), Pohang, South Korea, [3]Yahoo! Research, CA, USA

{pcy1302,jiyang3,qiz3,hanj}@illinois.edu,donghyun.kim@verizonmedia.com,hwanjoyu@postech.ac.kr

ABSTRACT

Network embedding is an influential graph mining technique for representing nodes in a graph as distributed vectors. However, the majority of network embedding methods focus on learning a single vector representation for each node, which has been recently criticized for not being capable of modeling multiple aspects of a node. To capture the multiple aspects of each node, existing studies mainly rely on offline graph clustering performed prior to the actual embedding, which results in the cluster membership of each node (i.e., node aspect distribution) fixed throughout training of the embedding model. We argue that this not only makes each node always have the same aspect distribution regardless of its dynamic context, but also hinders the end-to-end training of the model that eventually leads to the final embedding quality largely dependent on the clustering. In this paper, we propose a novel end-to-end framework for multi-aspect network embedding, called asp2vec, in which the aspects of each node are dynamically assigned based on its local context. More precisely, among multiple aspects, we dynamically assign a *single* aspect to each node based on its current context, and our aspect selection module is end-to-end differentiable via the Gumbel-Softmax trick. We also introduce the aspect regularization framework to capture the interactions among the multiple aspects in terms of relatedness and diversity. We further demonstrate that our proposed framework can be readily extended to heterogeneous networks. Extensive experiments towards various downstream tasks on various types of homogeneous networks and a heterogeneous network demonstrate the superiority of asp2vec.

CCS CONCEPTS

• **Computing methodologies** → *Unsupervised learning*.

KEYWORDS

Network Embedding, Representation Learning, Graph Mining

*Corresponding Author.

Figure 1: a) Clustering–based aspect assignment that fixes the aspect distribution during the embedding learning. b) asp2vec dynamically selects a single aspect based on the local context nodes.

1 INTRODUCTION

Networks constitute a natural paradigm to represent real-world relational data that contain various relationships between entities ranging from online social network of users, and academic publication network of authors, to protein-protein interaction (PPI) network in the physical world. Due to the pervasive nature of networks, analyzing and mining useful knowledge from networks has been an actively researched topic for the past decades. Among various tools for network analysis, *network embedding*, which learns continuous vector representations for nodes in a network, has recently garnered attention, and has been effectively and efficiently applied to various downstream network-based applications, such as node classification [5, 15, 18], and link prediction [1, 6].

A common underlying idea of network embedding methods is that a node embedding vector should be able to preserve the neighborhood structure of the node, i.e., local structural characteristics. Deepwalk [20] is a pioneering method that leverages the node co-occurrence information to learn the representations of nodes in a network such that nodes that frequently co-occur together have similar vector representations [8]. Specifically, the co-occurrence based methods usually perform random walks on a network to obtain node sequences on which the skip-gram [17] model is applied. However, past research on network embedding mostly assumes the existence of a single vector representation for each node, whereas in reality a node usually has multiple aspects. For example, nodes (e.g., authors) in an academic publication network often belong to multiple research communities, and thus modeling each node with a single vector entails information loss.

In this regard, several attempts have been recently made to model the multiple aspects of a node by learning multiple vector representations for each node [6, 15, 28]. However, there still remain several limitations. First, the aspects of the nodes are determined based on an offline clustering algorithm performed in advance (Figure 1a). In particular, to determine the node aspects, PolyDW [15] performs

clustering based on matrix factorization (MF) on the adjacency matrix, and similarly, Splitter [6] performs a local clustering algorithm based on ego-network analysis that encodes the role of a node in different local communities. However, as the clustering is done prior to the actual embedding learning, the cluster membership distribution (i.e., aspect distribution) for each node is fixed, which leads to each node always having the same aspect distribution regardless of its current context. This is especially problematic for the co-occurrence based network embedding methods, because the context of a node changes dynamically over multiple random walks. Moreover, owing to the offline clustering, the embedding methods cannot be trained in an end-to-end manner, which eventually leads to the final embedding quality largely dependent on the clustering.

Another limitation is that the interactions among the aspects are not explicitly captured. Although the interactions among aspects are implicitly revealed by the cluster membership of each node, it is only used for sampling the node aspects [6], and not directly incorporated for training the aspect embedding vectors. Going back to our example of academic publication network, authors can belong to multiple research communities, and these communities interact with each other. For example, data mining (DM) and database (DB) communities are more related to each other than data mining and computer architecture (CA) communities, and thus such interactions (i.e., relatedness: DM ↔ DB, and diversity: DM ↔ CA) should be captured. Furthermore, viewing the aspects in a broader perspective, we argue that the interactions among aspects differ according to the inherent characteristics of the networks. For example, nodes in an academic publication network tend to belong to fewer number of communities (i.e., aspects are inherently less diverse) than nodes in a PPI network. This is because an ordinary author in an academic publication network usually works on a limited number of research topics, whereas each protein in a PPI network is associated with various functions and involved in various interactions. As such, modeling the interactions among aspects is challenging, and thus should be done in a systematic way rather than implicitly by clustering.

In the light of these issues, we propose asp2vec, a novel end-to-end framework for multi-aspect network embedding. The core idea is to dynamically assign aspects to each node according to its local context as illustrated in Figure 1b (**Sec. 4.1**). More precisely, we selectively assign (sample) a *single* aspect for each node based on our assumption that each node should belong to a single aspect under a certain *local context*. For example, even an author with diverse research areas, who belongs to multiple research communities (i.e., multiple aspects), focuses on a single research topic when collaborating with a particular group of people (i.e., a single aspect within a local context). We materialize this idea by devising the *aspect selection module* based on the Gumbel-Softmax trick [12], which approximates the sampling from a categorical distribution in a differentiable fashion, thereby enabling an end-to-end training procedure (**Sec. 4.2**). Moreover, we introduce the *aspect regularization framework* to simultaneously capture the interactions and relationships among aspects in terms of both relatedness and diversity (**Sec. 4.3**). Finally, we demonstrate that our proposed framework can be readily extended to heterogeneous networks whose nodes and edges are multiple-typed (**Sec. 4.5**). Through our extensive experiments on 13 real-world datasets, including various types of homogeneous

networks and a heterogeneous network, we demonstrate the effectiveness of asp2vec in multiple downstream tasks including link prediction and author identification, in comparison with state-of-the-art multi-aspect network embedding methods. The source code of asp2vec can be found in https://github.com/pcy1302/asp2vec/.

2 RELATED WORK

Network embedding. Network embedding methods aim at learning low-dimensional vector representations for nodes in a graph while preserving the network structure [8, 20], and various other properties such as node attributes [18], and structural role [22]. More precisely, inspired by word2vec [17], DeepWalk [20] and node2vec [8] perform truncated random walks on graphs to obtain multiple node sequences, and then perform skip-gram to learn node embeddings by making an analogy between random walk sequences on a network and sentences in natural language. In another line of research, graph neural networks (GNNs) [30] recently have drawn intensive attention. Their main idea is to represent a node by aggregating information from its neighborhood [13, 26]. However, their performance largely depends on the available node labels [9, 13, 16], whereas our proposed framework is fully unsupervised. Recently, Velivckovic et al [27] proposed a GNN–based unsupervised network embedding method, called DGI, which maximizes the mutual information between the global representation of a graph and its local patches, i.e., node embeddings. However, while random walks naturally capture higher-order structural information and allow for the characterizing of different node contexts in networks, the performance of GNNs degenerates with larger number of convolution layers [14]. Moreover, unlike random walk–based algorithms [5], GNNs cannot be easily applied to heterogeneous networks without complicated design [18, 23].

Multi-aspect network embedding. While most network embedding methods focus on learning a single embedding vector for each node, several recent methods [6, 15, 16, 25, 28, 31] have been proposed to learn multiple embedding vectors for each node in a graph. More precisely, PolyDW [15] performs MF–based clustering to determine the aspect distribution for each node. Then, given random walk sequences on a graph, the aspect of the target and the context nodes are sampled independently according to the aspect distribution, and thus the target and the context nodes may have different aspects. Moreover, although capturing the local neighborhood structure [31] is more important for the co-occurrence based network embedding methods, PolyDW exploits clustering that focuses on capturing the global structure. Splitter [6] performs local clustering to split each node into multi-aspect representations, however, it blindly trains all the aspect embedding vectors to be close to the original node embedding vector rather than considering the aspect of the target node. MCNE [28] introduces a binary mask layer to split a single vector into multiple conditional embeddings, however, the number of aspects should be defined in the datasets (e.g., number of different types of user behaviors). Moreover, it formalizes the task of item recommendation as a supervised task, and combines network embedding methods with BPR [21], whereas our proposed framework is trained in a fully unsupervised manner with an arbitrary number of aspects. MNE [31] trains multi-aspect node embeddings based on MF, while considering the diversity of

the aspect embeddings. However, the aspect selection process is ignored, and the aspect embeddings are simply trained to be as dissimilar as possible to each other without preserving any relatedness among them.

Heterogeneous network embedding. A heterogeneous network (HetNet) contains multi-typed nodes and multi-typed edges, which should be modeled differently from the homogeneous counterpart, and there has been a line of research on heterogeneous network embedding [5, 7, 11]. Specifically, metapath2vec [5] extends Deepwalk by introducing a random walk scheme that is conditioned on meta-paths, where the node embeddings are learned by heterogeneous skip-gram. While various single aspect embedding methods have been proposed, multi-aspect network embedding is still in its infancy. More precisely, a line of work [3, 24] commonly projects nodes into multiple embedding spaces defined by multiple aspects, and the node embeddings are trained on each space. However, these methods define the aspects based on the edges whose ground-truth labels should be predefined (e.g., the edge between "Obama" and "Honolulu" is labeled as "was_born_in"), which is not always the case in reality. Finally, TaPEm [19] introduces the pair embedding framework that directly models the relationship between two nodes of different types, but still each node has a single embedding vector. Furthermore, although GNNs have also been recently applied to HetNets, they are either semi-supervised methods [29], or cannot be applied to a genuine HetNet whose both nodes and edges are multiple typed [18].

3 PRELIMINARY

Random Walk–based Unsupervised Network Embedding. Inspired by the recent advancement of deep learning methods, especially, skip-gram based word2vec [17], previous network representation learning methods [8, 20] started viewing nodes in a graph as words in a document. More precisely, these methods commonly perform a set of truncated random walks $\mathcal{W}$ on a graph, which are analogous to sequences of words, i.e., sentences. Then, the representation for each node is learned by optimizing the skip-gram objective, which is to maximize the likelihood of the context given the target node:

$$\max \sum_{\mathbf{w} \in \mathcal{W}} \sum_{v_i \in \mathbf{w}} \sum_{v_j \in \mathcal{N}(v_i)} \log \frac{\exp(\langle \mathbf{P}_i, \mathbf{Q}_j \rangle)}{\sum_{v_{j'} \in \mathcal{V}} \exp(\langle \mathbf{P}_i, \mathbf{Q}_{j'} \rangle)} \quad (1)$$

where $\mathbf{P}_i \in \mathbb{R}^d$ is the target embedding vector for node v_i, $\mathbf{Q}_j \in \mathbb{R}^d$ is the context embedding vector for node v_j, d is the number of embedding dimensions, $\langle a, b \rangle$ denotes a dot product between a and b, $\mathbf{w}$ is a node sequence, and $\mathcal{N}(v_i)$ denotes the nodes within the context window of node v_i. The final embedding vector for node v_i is generally obtained by averaging its target and context embedding vector. i.e., $(\mathbf{P}_i + \mathbf{Q}_i)/2$. By maximizing the above Eqn. 1, nodes that frequently appear together within a context window will be trained to have similar embeddings. Despite its effectiveness in learning node representations, it inherently suffers from a limitation that *each node is represented by a single embedding vector.*

3.1 Problem Statement

Definition 3.1. (**Multi-aspect Node Representation Learning**) Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, v_2, ..., v_n\}$ represents

the set of nodes, and $\mathcal{E} = \{e_{i,j}\}_{i,j}^n$ represents the set of edges, where $e_{i,j}$ connects v_i and v_j, we aim to derive a multi-aspect embedding matrix $\mathbf{Q}_i \in \mathbb{R}^{K \times d}$ for each node v_i, where K is an arbitrary predefined number of aspects. More precisely, each aspect embedding vector $\{\mathbf{Q}_i^{(s)} \in \mathbb{R}^d\}_{s=1}^K$ should 1) preserve the network structure information, 2) capture node v_i's different aspects, and 3) model the interactions among the aspects (i.e., relatedness and diversity).

4 PROPOSED FRAMEWORK: ASP2VEC

We present our proposed framework for context–based multi–aspect network embedding (Sec. 4.1) that includes the aspect selection module (Sec. 4.2) and the aspect regularization framework (Sec. 4.3). Then, we demonstrate how asp2vec can be further extended to a heterogeneous network (Sec 4.5). Figure 2 summarizes the overall architecture of our proposed framework, called asp2vec.

4.1 Context–based Multi–aspect Network Embedding

Given a target node and its local context, the core idea is to first determine the current aspect of the target node based on its local context, and then predict the context node with respect to the selected aspect of the target node. More precisely, given a target node embedding vector $\mathbf{P}_i \in \mathbb{R}^d$, and its currently selected aspect $\delta(v_i) \in \{1, 2, ..., K\}$, our goal is to predict its context embedding vectors with respect to the selected aspect $\delta(v_i)$. i.e., $\{\mathbf{Q}_j^{(\delta(v_i))}|v_j \in \mathcal{N}(v_i)\}$. Formally, for each random walk sequence $\mathbf{w} \in \mathcal{W}$, we aim to maximize the following objective:

$$\mathcal{J}_{\text{asp2vec}}^{(\mathbf{w})} = \sum_{v_i \in \mathbf{w}} \sum_{v_j \in \mathcal{N}(v_i)} \sum_{s=1}^K p(\delta(v_i) = s) \log p(v_j | v_i, p(\delta(v_i) = s))$$

$$= \sum_{v_i \in \mathbf{w}} \sum_{v_j \in \mathcal{N}(v_i)} \sum_{s=1}^K p(\delta(v_i) = s) \log \frac{\exp(\langle \mathbf{P}_i, \mathbf{Q}_j^{(s)} \rangle)}{\sum_{v_{j'} \in \mathcal{V}} \exp(\langle \mathbf{P}_i, \mathbf{Q}_{j'}^{(s)} \rangle)} \quad (2)$$

where $\mathbf{Q}_j^{(s)} \in \mathbb{R}^d$ is the embedding vector of node v_j in terms of aspect s, $p(\delta(v_i) = s)$ denotes the probability of v_i being selected to belong to the aspect s, where $\sum_{s=1}^K p(\delta(v_i) = s) = 1$, and $p(v_j | v_i, p(\delta(v_i) = s))$ denotes the probability of a context node v_j given a target node v_i whose aspect is s. Note that this is in contrast to directly maximizing the probability of its context node regardless of the aspect of the target node as in Eqn. 1.

4.2 Determining the Aspect of the Center Node

Next, we introduce the *aspect selection module* that computes the aspect selection probability $p(\delta(v_i))$. Here, we assume that the aspect of each target node v_i can be readily determined by examining its local context $\mathcal{N}(v_i)$. i.e., $p(\delta(v_i)) \equiv p(\delta(v_i)|\mathcal{N}(v_i))$. More precisely, we apply softmax to model the probability of v_i's aspect:

$$p(\delta(v_i) = s) \equiv p(\delta(v_i) = s|\mathcal{N}(v_i)) = \text{softmax}(\langle \mathbf{P}_i, \text{Readout}^{(s)}(\mathcal{N}(v_i)) \rangle)$$

$$= \frac{\exp(\langle \mathbf{P}_i, \text{Readout}^{(s)}(\mathcal{N}(v_i)) \rangle)}{\sum_{s'=1}^K \exp(\langle \mathbf{P}_i, \text{Readout}^{(s')}(\mathcal{N}(v_i)) \rangle)} \quad (3)$$

where we leverage a readout function $\text{Readout}^{(s)} : \mathbb{R}^{|\mathcal{N}(v_i)| \times d} \to \mathbb{R}^d$ to summarize the information encoded in the local context of node v_i, i.e., $\mathcal{N}(v_i)$, with respect to aspect s.
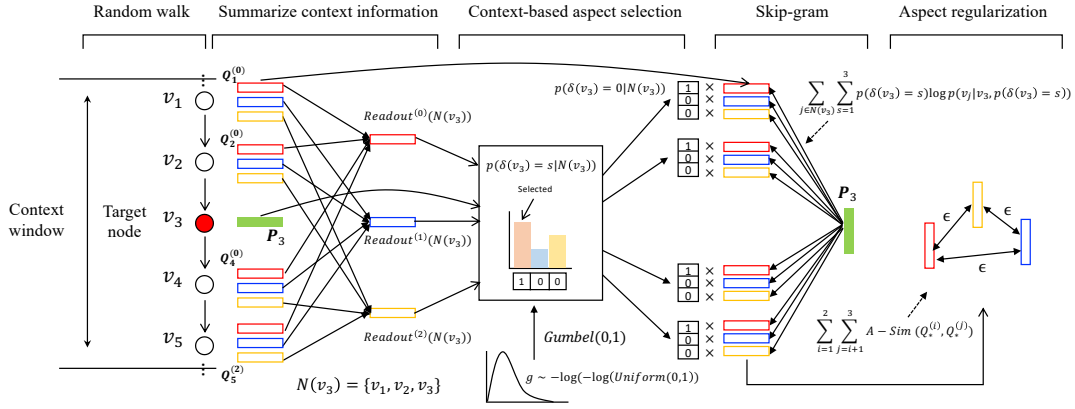
Figure 2: The overall architecture of asp2vec. Given a sequence of nodes within a context window, a single aspect is dynamically selected according to the aspect of the context nodes. Then, the skip-gram is performed based on the selected aspect. The aspect regularization framework encourages the diversity and preserves the relatedness among the aspect embedding vectors.

4.2.1 ***Gumbel-Softmax based Aspect Selection*** *.* Note that modeling the aspect selection probability $p(\delta(v_i))$ based on softmax (Eqn. 3) assigns a probability distribution over the number of aspects K, which is *continuous*. However, we argue that although a node may simultaneously belong to multiple aspects in a global perspective, *it should be assigned to a single aspect within a local context.* For example, even an author with diverse research areas, who belongs to multiple research communities, focuses on a single research topic when collaborating with a particular group of people. In other words, the aspect selection should be done in a discrete manner based on a categorical distribution over K aspects. i.e., hard selection. However, since the hard selection is a non-differentiable operation, it blocks the flow of the gradients down to the embedding vectors, and prevents the end-to-end back-propagation optimization. As a workaround, we adopt the Gumbel-Softmax trick [12], which approximates the sampling from a categorical distribution by reparameterizing the gradients with respect to the parameters of the distribution. More precisely, given a K-dimensional categorical distribution with class probability $\pi_1, \pi_2, .., \pi_K$, the gumbel-max trick provides a simple way to draw a one-hot sample $z = (z_1, \cdots z_K) \in \mathbb{R}^K$ from the categorical distribution as follows:

$$z = \text{one-hot} \left(\text{argmax}_i \left[\log \pi_i + g_i \right] \right) \quad (4)$$

where g_i is the gumbel noise drawn from Gumbel(0,1) distribution, which can be obtained as follows:

$$g_i = -\log(-\log(u_i)) \qquad u_i \sim Uniform(0, 1) \quad (5)$$

Due to the non-differentiable nature of the argmax($\cdot$) operation in Eqn. 4, we further approximate it by using softmax to ensure the differentiability as follows, which is known as the Gumbel-Softmax trick [12]:

$$z_i = \text{softmax} \left[\log \pi_i + g_i \right]$$
$$= \frac{\exp \left((\log \pi_i + g_i) / \tau \right)}{\sum_{j=1}^{K} \exp \left((\log \pi_j + g_j) / \tau \right)} \quad \text{for } k = 1, \ldots, K \quad (6)$$

where τ is the temperature parameter to control the extent to which the output approximates the argmax($\cdot$) operation: As $\tau \to 0$, samples from the Gumbel-Softmax distribution become one-hot. Finally, replacing the softmax in Eqn. 3 with the Gumbel-Softmax in Eqn. 6,

we obtain the following aspect selection probability:

$$p(\delta(v_i) = s \mid N(v_i)) = \frac{\exp[\langle \mathbf{P}_i, \text{Readout}^{(s)}(N(v_i))\rangle + g_s] / \tau}{\sum_{s'=1}^{K} \exp[\langle \mathbf{P}_i, \text{Readout}^{(s')}(N(v_i))\rangle + g_{s'}] / \tau} \quad (7)$$

Combining the loss for all walks $\mathbf{w} \in \mathcal{W}$ described in Eqn. 2, the minimization objective for learning the context-based multi-aspect node representation is given by:

$$\mathcal{L}_{\text{asp2vec}} = - \sum_{\mathbf{w} \in \mathcal{W}} \mathcal{J}_{\text{asp2vec}}^{(\mathbf{w})} \quad (8)$$

4.2.2 ***Readout Function***. Recall that we expect the output of Readout$^{(s)}(N(v_i))$ to capture the current aspect s of the target node v_i by leveraging its local context information $N(v_i)$. Among various choices for Readout($\cdot$), we choose the average pooling operation:

$$\text{Readout}^{(s)}(N(v_i)) = \frac{1}{|N(v_i)|} \sum_{v_j \in N(v_i)} Q_j^{(s)} = \bar{Q}_{N(v_i)}^{(s)} \quad (9)$$

It is worth noting that thanks to its simplicity and efficiency, we choose the average pooling over the more advanced techniques that assign different weights to each context node, such as recurrent neural network–based or attention–based pooling [33].

Discussion. If each node is associated with some attribute information, determining the aspect of each node can be more intuitive than solely relying on the context node embeddings originally learned to preserve the graph structure. In this regard, we study in Sec. 4.5.2 how to model the aspect when node attributes are given.

4.3 Modeling Relationship Among Aspects

As different aspect embeddings are intended to capture different semantics of each node, they should primarily be sufficiently diverse among themselves. However, at the same time, these *aspects are inherently related to each other* to some extent. For example, in an academic publication network, each node (i.e., author) may belong to various research communities (i.e.aspects), such as DM, CA, and DB community. In this example scenario, we can easily tell that DM and DB communities are more related than DM and CA communities, which shows the importance of modeling the interaction among aspects. In short, aspect embeddings should not only be 1) **diverse** (e.g., DM↔CA) so as to independently capture

the inherent properties of individual aspects, but also 2) **related** (e.g., DM↔DB) to each other to some extent so as to capture some common information shared among aspects.

To this end, we introduce a novel aspect regularization framework, called reg$_{\text{asp}}$, which is given by:

$$\text{reg}_{\text{asp}} = \sum_{i=1}^{K-1} \sum_{j=i+1}^{K} \text{A-Sim}(Q_*^{(i)}, Q_*^{(j)}) \tag{10}$$

where $Q_*^{(i)} \in \mathbb{R}^{n \times d}$ is the aspect embedding matrix in terms of aspect i for all nodes $v \in \mathcal{V}$; $|\mathcal{V}| = n$, and A-Sim$(\cdot, \cdot)$ measures the similarity score between two aspect embedding matrices: large A-Sim$(Q_*^{(i)}, Q_*^{(j)})$ refers to the two aspects sharing some information in common, whereas small A-Sim$(Q_*^{(i)}, Q_*^{(j)})$ means aspect i and aspect j capture vastly different properties. The aspect similarity score A-Sim$(Q_*^{(i)}, Q_*^{(j)})$ is computed by the sum of the similarity scores between aspect embeddings of each node:

$$\text{A-Sim}(Q_*^{(i)}, Q_*^{(j)}) = \sum_{h=1}^{|V|} f(Q_h^{(i)}, Q_h^{(j)}) \tag{11}$$

where $Q_h^{(i)} \in \mathbb{R}^d$ is the embedding vector of node v_h in terms of aspect i, and $f(Q_h^{(i)}, Q_h^{(j)})$ denotes the aspect similarity score (ASS) between embeddings of aspects i and j with respect to node v_h. By minimizing reg$_{\text{asp}}$ in Eqn. 10, we aim to learn *diverse* aspect embeddings capturing inherent properties of each aspect. In this work, we evaluate the ASS based on the cosine similarity between two aspect embedding vectors given by the following equation:

$$f(Q_h^{(i)}, Q_h^{(j)}) = \frac{\langle Q_h^{(i)}, Q_h^{(j)} \rangle}{\|Q_h^{(i)}\| \|Q_h^{(j)}\|}, \quad -1 \le f(Q_h^{(i)}, Q_h^{(j)}) \le 1 \tag{12}$$

However, as mentioned earlier, the aspect embeddings should not only be diverse but also to some extent *related* to each other (e.g., DM↔DB). In other words, as the aspects are not completely independent from each other, we should model their interactions. To this end, we introduce a binary mask $w_{i,j}^h$ to selectively penalize the aspect embedding pairs according to their ASSs [2]. More precisely, the binary mask is defined as:

$$w_{i,j}^h = \begin{cases} 1, & \left| f(Q_h^{(i)}, Q_h^{(j)}) \right| \ge \epsilon \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

where ϵ is a threshold parameter that controls the amount of information shared between a pair of aspect embedding vectors of a given node: large ϵ encourages the aspect embeddings to be related to each other, whereas small ϵ encourages diverse aspect embeddings. Specifically, for a given node v_h, if the absolute value of ASS between the aspect embedding vectors of aspect i and j, i.e., $\left| f(Q_h^{(i)}, Q_h^{(j)}) \right|$, is greater than ϵ, then we would like to penalize, and accept it otherwise. In other words, we allow two aspect embedding vectors of a node to be similar to each other to some extent (as much as ϵ). By doing so, we expect that the aspect embeddings to be sufficiently diverse, but at the same time share some information in common.

Note that the aspect regularization framework is more effective for networks with diverse aspects. For example, users in social networks tend to belong to multiple (diverse) communities compared

with authors in academic networks, because authors usually work on a limited number of research topics. Therefore, as the aspect regularization framework encourages the aspect embeddings to be diverse, it is expected to be more effective in social networks than in academic networks. We later demonstrate that this is indeed the case, which implies that asp2vec can also uncover the hidden characteristics of networks. We consider both positive and negative similarities by taking an absolute value of the cosine similarity, i.e., $|f(\cdot, \cdot)|$, because negative similarity still means the two aspects are semantically related yet in an opposite direction.

Armed with the binary mask defined in Eqn. 13, Eqn. 11 becomes:

$$\text{A-Sim}(Q_*^{(i)}, Q_*^{(j)}) = \sum_{h=1}^{|V|} w_{i,j}^h f(Q_h^{(i)}, Q_h^{(j)}) \tag{14}$$

The final objective function is to jointly minimize the context–based multi-aspect network embedding loss $\mathcal{L}_{\text{asp2vec}}$ in Eqn. 8, and the aspect regularization loss reg$_{\text{asp}}$ in Eqn. 10:

$$\mathcal{L} = \mathcal{L}_{\text{asp2vec}} + \lambda \, \text{reg}_{\text{asp}} \tag{15}$$

where λ is the coefficient for the aspect regularization.

Discussions on the Number of Parameters. A further appeal of our proposed framework is its superior performance with a relatively limited number of parameters. More precisely, PolyDW [15], which is one of the most relevant competitors, requires $2|V|dK$ parameters for node embeddings ($|V|dK$ each for target and context) in addition to the parameters required for the offline clustering method. i.e., matrix factorization, which requires another $2|V|d$, thus $2|V|d(K+1)$ in total. Moreover, Splitter [6] requires $|V|d(K+2)$ for node embeddings ($2|V|d$ for Deepwalk parameters and $|V|dK$ for persona embedding), and additional parameters for local and global clustering. On the other hand, our proposed framework only requires $|V|d(K+1)$ for node embeddings ($|V|d$ for target and $|V|dK$ for context) without additional parameters for clustering. Hence, we argue that asp2vec outperforms state-of-the-art multi-aspect network embedding methods with less parameters.

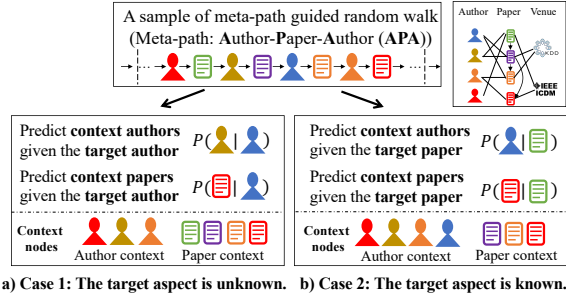## 4.4 Task: Link Prediction

To evaluate the performance of our framework, we perform link prediction, which is to predict the strength of the linkage between two nodes. Link prediction is the best choice for evaluating the multi-aspect network embedding methods, because aspect embeddings are originally designed to capture various interactions among nodes, such as membership to multiple communities within a network, which is best revealed by the connection information. Moreover, link prediction is suggested by recent work as the primary evaluation task for unsupervised network embedding methods compared with node classification that involves a labeling process that may be uncorrelated with the graph itself [1, 6].

Recall that in our proposed framework, a node v_i has a center embedding vector $P_i \in \mathbb{R}^d$, and K different aspect embedding vectors $\{Q_i^{(s)} \in \mathbb{R}^d\}_{s=1}^{K}$, which add up to $K + 1$ embedding vectors in total for each node. In order to obtain the final embedding vector for node v_i, we first compute the average of the aspect embedding vectors, and add it to the center embedding vector:

$$U_i = P_i + \frac{1}{K} \sum_{s=1}^{K} Q_i^{(s)} \tag{16}$$

a) Case 1: The target aspect is unknown. b) Case 2: The target aspect is known.

Figure 3: Two different cases should be considered for asp2vec-het.

where $\mathbf{U}_i \in \mathbb{R}^d$ is the final embedding vector for node v_i. Note that in previous work [6, 15] that learn multiple embedding vectors for each node, the final link prediction is done by calculating the sum [15] or the maximum [6] of the cosine similarity between all possible pairs of aspect embedding vectors. Both require $O(|\mathcal{V}|^2 K^2)$ dot product operations to compute the link probability between all pairs of nodes, which is time consuming. In this work, we simply use the final embedding vector $\mathbf{U}_i$ on which any off-the-shelf classification algorithm, such as logistic regression, is trained facilitating more practical usage in the real-world.

4.5 Extension to Heterogeneous Network

Heretofore, we described our approach for learning context–based multi–aspect node representations for a network with a a single type of nodes and a single type of edges. i.e., homogeneous network. In this section, we demonstrate that our proposed multi-aspect network embedding framework, asp2vec, can be readily extended to learn representations for nodes in a HetNet. Note that PolyDW [15] also showed its extendability to a HetNet. However, it is only limited to a bipartite network without node attribute information, whereas asp2vec can be extended to any type of HetNets with node attribute information.

Recall that in Sec. 4.4 we defined our primary task as link prediction. In this regard, among various link prediction tasks that can be defined in a HetNet such as, recommendation [10] (i.e., user-item), and author identification (i.e., paper-author), we specifically focus on the task of author identification in big scholarly data whose task is to predict the true authors of an anonymized paper [4, 19, 32], i.e., link probability between paper-typed nodes and author-typed nodes. Note that each paper is assumed to be associated with its content information. i.e., abstract text.

4.5.1 Context–based Multi–aspect Heterogeneous Network Embedding.
To begin with, we first perform meta-path guided random walk [5] to generate sequences of nodes, which is an extension of the truncated random walk [8, 20] to a HetNet. More precisely, random walks are conditioned on a meta-path $\mathcal{P}$ [5], which is a composition of relations that contains a certain semantic. Figure 3 shows an example metapath "Author-Paper-Author (APA)" that refers to a co-authorship relation. After generating a set of random walks $\mathcal{W}_{\mathcal{P}}$ guided by a meta-path $\mathcal{P}$, we aim to maximize the probability of context nodes given a target node similar to Eqn. 2. However, unlike the former case, a walk $\mathbf{w} \in \mathcal{W}_{\mathcal{P}}$ now contains multiple types of nodes. Hence, we propose asp2vec-het by revising

Eqn. 2 to incorporate the heterogeneity as follows:

$$(17)$$
$$\mathcal{J}_{\text{asp2vec-het}}^{(\mathbf{w})} = \sum_{v_i \in \mathbf{w}} \sum_{t \in \mathcal{T}_{\mathcal{V}}} \sum_{v_j \in \mathcal{N}_t(v_i)} \sum_{s=1}^{K} p(\delta(v_i) = s) \log p(v_j | v_i, p(\delta(v_i) = s))$$

where $\mathcal{T}_{\mathcal{V}}$ denotes the set of node types (e.g., author, paper, and venue), and $\mathcal{N}_t(v_i)$ denotes node v_i's context nodes with type t.

In order to apply asp2vec to a HetNet, we need to consider the following two cases and model them differently: 1) case 1: the aspect of the target node is unknown, thus should be inferred, and 2) case 2: the aspect of the target node is already revealed, thus there is no need to infer its aspect. Concretely, taking a sample of random walk guided by meta-path APA (Figure 3), case 1 is when the target node is author, and case 2 is when the target node is paper. Specifically, in case 2 shown in Figure 3b, we argue that the aspect (i.e. topic) of a paper can be easily inferred by looking at its content (i.e., text), whereas in case 1 shown in Figure 3a, the aspect of an author should still be inferred from its context. Therefore, case 1 should be modeled by our asp2vec-het framework in Eqn. 17, and case 2 can be simply modeled by the previous single aspect embedding method, such as metapath2vec [5] as follows:

$$\mathcal{J}_{\text{HetDW}}^{(\mathbf{w})} = \sum_{v_i \in \mathbf{w}} \sum_{t \in \mathcal{T}_{\mathcal{V}}} \sum_{v_j \in \mathcal{N}_t(v_i)} \log p(v_j | v_i) \qquad (18)$$

The final objective function is to jointly optimize Eqn. 17 (case 1) and Eqn. 18 (case 2) in addition to the aspect regularization framework in Eqn. 10 as follows:

$$\mathcal{L}_{\text{Het}} = - \sum_{\mathcal{P} \in \mathcal{S}(\mathcal{P})} \sum_{\mathbf{w} \in \mathcal{W}_{\mathcal{P}}} \left[\mathcal{J}_{\text{asp2vec-het}}^{(\mathbf{w})} + \mathcal{J}_{\text{HetDW}}^{(\mathbf{w})} \right] + \lambda \, \text{reg}_{\text{asp}} \qquad (19)$$

where $\mathcal{S}(\mathcal{P})$ denotes all predefined meta-path schemes.

Note that although we specifically described the scenario in the author identification for the ease of explanations, asp2vec-het can be easily applied to other tasks such as recommendation, where the task is to infer the user-item pairwise relationship.

4.5.2 Determining node aspect in a HetNet.
Recall that in Sec. 4.2, given a homogeneous network, we determine the aspect of a target node based on its context nodes. In a HetNet, the node aspects can be similarly determined, however, we should take into account the heterogeneity of the node types among the context nodes. In other words, some node types might be more helpful than others in terms of determining the aspect of the target node. For example, for a target node "author" in Figure 3, a context node whose type is "paper" is more helpful than another author-typed node because a paper is usually written about a single topic, whereas an author can work on various research areas. Moreover, it is important to note that determining the aspect of the target node becomes even more straightforward if node attribute information is provided, for example, the content of the paper. In this regard, we leverage the paper abstract information to obtain the paper embedding vector, and rely only on the paper nodes to determine the aspect of the target author node. Specifically, we employ a GRU-based paper encoder introduced in [32] that takes pretrained word vectors of an abstract, and returns a single vector that encodes the paper content. Refer to Section 3.1 of [32] for more detail.

Table 1: The overall performance for link prediction in terms of AUC-ROC (OOM: Out of memory).

dim ($d \times K$)	100 ($d = 20$, $K = 5$)					200 ($d = 40$, $K = 5$)					500 ($d = 100$, $K = 5$)				
	DW	DGI	PolyDW	Splitter	asp2vec	DW	DGI	PolyDW	Splitter	asp2vec	DW	DGI	PolyDW	Splitter	asp2vec
Filmtrust	0.6850	0.6973	0.6953	0.6128	**0.7426**	0.7399	0.7094	0.6841	0.6111	**0.7460**	0.7415	0.7215	0.6643	0.6097	**0.7501**
Wiki-vote	0.6273	0.5860	0.5557	0.5190	**0.6478**	0.6277	0.5741	0.5179	0.5085	**0.6464**	0.6260	**0.6540**	0.5161	0.5048	0.6507
CiaoDVD	0.7136	0.6809	0.6528	0.5978	**0.7430**	0.7014	0.6696	0.6263	0.5881	**0.7447**	0.7140	0.6897	0.6058	0.5819	**0.7450**
BlogCatalog	0.8734	0.9191	0.7505	0.8441	**0.9503**	0.9220	0.9083	0.6944	0.8199	**0.9548**	0.9331	OOM	0.6249	0.7876	**0.9429**
Epinions	0.7188	0.6684	0.7038	0.6880	**0.7416**	0.7223	0.6711	0.6884	0.6733	**0.7441**	0.7312	OOM	0.6720	0.6581	**0.7459**
Flickr	0.9506	0.9214	0.9146	0.9528	**0.9584**	0.9580	OOM	0.8862	0.8582	0.9571	0.9570	OOM	0.8582	0.9299	**0.9678**
PPI	0.8236	0.8087	0.7286	0.8372	**0.8887**	0.8237	0.8341	0.6995	0.8346	**0.8947**	0.8214	0.8593	0.6693	0.8336	**0.8991**
Wikipedia	0.7729	0.8984	0.6259	0.6897	**0.9049**	0.8677	0.8927	0.5920	0.6939	**0.9040**	0.8414	**0.9029**	0.5218	0.7018	0.9011
Cora	**0.9181**	0.8223	0.8504	0.8357	0.8814	**0.9110**	0.8300	0.8416	0.8361	0.9056	0.8814	**0.9475**	0.8393	0.8412	0.9181
ca-HepTh	**0.9080**	0.8661	0.8806	0.8827	0.8989	**0.9160**	0.8787	0.8812	0.9076	0.9119	**0.9219**	0.7402	0.8831	0.9058	0.9185
ca-AstroPh	**0.9784**	0.9144	0.9661	0.9731	0.9734	0.9803	0.9690	0.9734	0.9791	**0.9821**	0.9775	OOM	0.9754	0.9827	**0.9842**
4area	**0.9548**	0.9253	0.9441	0.9355	0.9503	0.9551	0.9349	0.9449	0.9496	**0.9587**	0.9553	OOM	0.9463	0.9550	**0.9627**

Table 2: Statistics of the datasets. (Dir.: directed graph.)

		Dataset	Num. nodes	Num. edges
Homogeneous Network	Social Network	Filmtrust (Dir.)	1,642	1,853
		Wiki-vote (Dir.)	7,066	103,689
		CiaoDVD (Dir.)	7,375	111,781
		BlogCatalog	10,312	333,983
		Epinions (Dir.)	49,290	487,181
		Flickr	80,513	5,899,882
		PPI	3,890	76,584
		Wikipedia (Word co-occurrence)	4,777	184,812
	Academic Network	Cora	2,708	5,429
		ca-HepTh	9,877	25,998
		ca-AstroPh	18,772	198,110
		4area	27,199	66,832
HetNet	DBLP	Num. authors	Num. papers	Num. venues
		27,920	21,808	18

5 EXPERIMENTS

Datasets. We evaluate our proposed framework on **thirteen** commonly used publicly available datasets including a HetNet. The datasets can be broadly categorized into social network, protein network, word-co-occurrence network, and academic network. Table 2 shows the statistics of the datasets used in our experiments. Refer to the appendix for more detail.

Methods Compared. As asp2vec is an unsupervised multi-aspect network embedding framework that can be applied on both homogeneous and heterogeneous networks, we compare with the following unsupervised methods:

(1) Unsupervised embedding methods for homogeneous networks
- Models a single aspect
  - **Deepwalk** [20]/**node2vec** [8]: They learn node embeddings by random walks and skip-gram. As they generally show similar performance, we report the best performing method among them.
  - **DGI** [27]: It is the state-of-the-art unsupervised network embedding method that maximizes the mutual information between the graph summary and the local patches of a graph.
- Models multiple aspects
  - **PolyDW** [15]: It performs MF–based clustering to obtain aspect distribution for each node from which an aspect for the target and the context nodes are independently sampled.
  - **Splitter** [6]: It splits each node into multiple embeddings by performing local graph clustering, called persona2vec.

(2) Unsupervised embedding methods for heterogeneous networks
- Models a single aspect

  - **M2V++** [5]: It learns embeddings for nodes in a HetNet by performing meta-path guided random walk followed by heterogeneous skip-gram. Following [19], we leverage the paper abstract for paper embeddings.
  - **Camel** [32]: It is a task-guided heterogeneous network embedding method developed for the author identification task in which content-aware skip-gram is introduced.
- Models multiple aspects
  - **TaPEm** [19]: It is the state-of-the-art task-guided heterogeneous network embedding method that introduces the pair embedding framework to directly capture the pairwise relationship between two heterogeneous nodes.
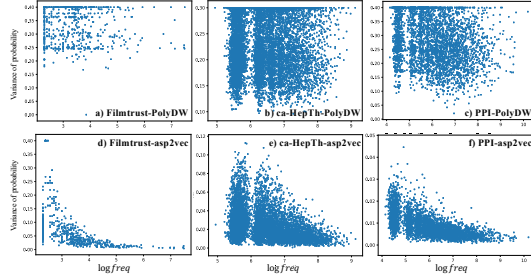
Experimental Settings and Evaluation Metrics. For evaluations of asp2vec and asp2vec-het, we perform link prediction and author identification, respectively. For link prediction, we follow the protocol of Splitter [1, 6], and for author identification, we follow the protocol of [19, 32]. As for the evaluation metrics, we use AUC-ROC for link prediction, and recall, F1, and AUC for author identification. Note that for all the tables in this section, the number of aspects K is 1 for all the single aspect embedding methods, i.e. DW, DGI, M2V++, Camel, and TaPEm. For more information regarding the hyperparameters and the model training–related details, refer to the appendix.

5.1 Performance Analysis

Overall Evaluation. Table 1 shows the link prediction performance of various methods. We have the following observations: **1)** asp2vec generally performs well on all datasets, especially outperforming other methods on social networks, PPI and Wikipedia networks. We attribute such behavior to the fact that nodes in these networks inherently exhibit multiple aspects compared with nodes in the academic networks, where each node is either a paper (Cora) or an author (ca-HepTh, ca-AstroPh, and 4area) that generally focuses on a single research topic, thus we have less distinct aspects to be captured. In particular, nodes (authors) in ca-AstroPh and ca-HepTh networks are even more focused as they are specifically devoted to Astro Physics and High Energy Physics Theory, respectively. On the other hand, social networks contain various communities, for example, BlogCatalog is reported to have 39 ground truth groups in the network. **2)** We observe that the prediction performance on the academic networks are generally high for all methods, which aligns with the results reported in [6] where DW generally performs on par with Splitter under the

Table 3: Benefit of the Gumbel-Softmax.

$d = 20, K = 5$	Softmax	Gumbel-Softmax	Improvement
Filmtrust	0.6421	**0.7426**	15.65%
Wiki-vote	0.6165	**0.6478**	5.08%
CiaoDVD	0.6162	**0.7430**	20.58%
BlogCatalog	0.7323	**0.9503**	29.77%
Epinions	0.6693	**0.7416**	10.80%
Flickr	0.8956	**0.9584**	7.01%
PPI	0.6919	**0.8887**	28.44%
Wikipedia	0.8269	**0.9049**	9.43%
Cora	0.8605	**0.8814**	2.43%
ca-HepTh	0.8890	**0.8989**	1.11%
ca-AstroPh	0.9116	**0.9734**	6.78%
4area	0.9286	**0.9503**	2.34%

Table 4: Link prediction performance (AUC-ROC) without reg_{asp}, and over various thresholds (ϵ).

dim = 100	No	Threshold (ϵ)					best vs.
($d = 20, K = 5$)	reg_{asp}	0.9	0.7	0.5	0.3	0.1	No reg_{asp}
Filmtrust	0.660	**0.743**	0.742	0.740	0.738	0.735	12.58%
Wiki-vote	0.616	0.647	**0.648**	0.647	0.647	0.645	5.15%
CiaoDVD	0.617	**0.743**	0.742	0.742	0.738	0.735	20.37%
BlogCatalog	0.791	0.948	**0.950**	0.949	0.939	0.869	20.11%
Epinions	0.684	**0.742**	0.741	0.738	0.731	0.693	8.37%
Flickr	0.897	0.955	**0.958**	0.954	0.954	0.929	6.85%
PPI	0.729	0.880	0.885	**0.889**	0.881	0.819	21.97%
Wikipedia	0.841	0.896	0.904	**0.905**	0.880	0.850	7.60%
Cora	0.879	0.881	0.880	**0.881**	0.862	0.857	0.23%
ca-HepTh	0.879	**0.899**	0.896	0.898	0.893	0.864	2.30%
ca-AstroPh	0.921	**0.973**	0.973	0.971	0.967	0.939	5.56%
4area	0.919	**0.950**	0.949	0.946	0.940	0.915	3.44%

Figure 4: Variance of the aspect selection probability over the frequency of the node appearing in random walks (Top row: PolyDW, Bottom row: asp2vec).

figure represents the variance of the aspect distribution to which each target node is assigned. As expected, asp2vec (Figure 4 bottom) tends to assign low variance aspect distribution to high frequency nodes, and low frequency nodes have high variance aspect distribution. On the other hand, as PolyDW determines aspects by an offline clustering algorithm, there is no such tendency observed (Figure 4 Top), which verifies the superiority of our context–based aspect selection module.

same embedding size. We conjecture that this is because links in academic networks are relatively easy to predict because academic research communities are relatively focused, thereby nodes having less diverse aspects as mentioned above. **3)** We observe that DGI generally performs better as the embedding size increases, outperforming others in some datasets when $d = 500$. However, DGI is not scalable to large dimension sizes, which is also mentioned in the original paper [27] (DGI fixed $d = 512$ for all datasets, but due to memory limitation, d is reduced to 256 for pubmed dataset that contains 19,717 nodes).
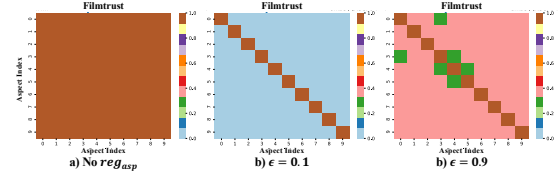
Benefit of Aspect Selection Module. Table 3 shows the comparisons between asp2vec with conventional softmax (Eqn. 3), and the Gumbel-Softmax trick (Eqn. 7). We found that the Gumbel-Softmax indeed is beneficial, and more importantly, we observe that the improvements are more significant for social networks, PPI, and Wikipedia network, compared with the academic networks. This verifies that the aspect modeling is more effective for networks with inherently diverse aspects, such as social and PPI networks.

To further verify the benefit of the aspect selection module compared with the offline clustering–based method, we examine how the aspects are actually assigned to nodes. Our assumption is nodes that frequently appear in random walks (i.e., $\mathcal{W}$) are more likely to exhibit more aspects compared with the less frequently appearing ones. Therefore, the variance of aspect probability distribution of a frequently appearing node will be relatively smaller than that of the less frequently appearing one. For example, given four aspects, a frequently appearing node may have an aspect probability distribution [0.2,0.3,0.3,0.2], whereas that of a less frequently appearing node is likely to have a skewed distribution that looks like [0.7,0.0,0.3,0.0] with a higher variance. In Figure 4, we plot the variance of the aspect probability distribution of each target node according to its frequency within the random walks. Precisely, each dot in the

Figure 5: Heatmaps showing the similarity between the aspect embeddings (Best viewed in color).

Benefit of Aspect Regularizer. Table 4 shows the link prediction AUC-ROC over various thresholds (ϵ) for reg_{asp}. **1)** We observe that the performance drops significantly when $\epsilon = 1$, that is, when the aspect regularization framework is not incorporated (no reg_{asp}), which verifies the benefit of the aspect regularization framework. **2)** The aspect regularization framework is less effective on the academic networks (see best vs. No reg_{asp}). Again, this is because academic networks inherently have less diverse aspects, which reduce the need for modeling diverse aspect embeddings. **3)** In Figure 5, we illustrate heatmaps where each element denotes the cosine similarity between a pair of aspect embedding vectors. More precisely, we compute the pairwise similarity between aspect embedding vectors for all nodes, and then compute their average to obtain the final heatmap. We observe that the aspect embeddings are trained to be highly similar to each other without reg_{asp} (Figure 5a), which shows the necessity of the aspect regularization framework. Moreover, when $\epsilon = 0.1$ (Figure 5b), the similarity among the aspect embeddings is small, which demonstrates that a small ϵ encourages the aspect embeddings to be diverse. Finally, when $\epsilon = 0.9$ (Figure 5c), we observe that some of the aspect embeddings are trained to be less similar (i.e. green) than others (i.e. red) allowing more flexibility in learning the aspect embeddings.

Evaluations on HetNet. Table 5 shows the result of author identification. **1)** We observe that asp2vec-het significantly outperforms the state-of-the-art task-guided author identification methods in all

Table 5: Author identification results in terms of recall, F1, and AUC over various dimensions ($d \times K$).

dim	Methods	Recall@N		F1@N		AUC
		5	10	5	10	
200	M2V++	0.4607	0.6115	0.2691	0.2100	0.8752
	Camel	0.5774	0.7270	0.3384	0.2518	0.9021
	TaPEm	0.6150	0.7473	0.3593	0.2586	0.9031
	asp2vec-het	**0.6474**	**0.7701**	**0.3832**	**0.2709**	**0.9137**
500	M2V++	0.4137	0.5662	0.2442	0.1970	0.8433
	Camel	0.4788	0.6243	0.2827	0.2197	0.8602
	TaPEm	0.5193	0.6592	0.3058	0.2319	0.8643
	asp2vec-het	**0.6601**	**0.7671**	**0.3879**	**0.2693**	**0.9161**

the metrics, which verifies the benefit of our context–based aspect embedding framework. More precisely, asp2vec-het outperforms TaPEm [19], which is a recently proposed method whose motivation is similar to ours in that it models multiple aspects of each node by the pair embedding framework. This verifies that explicitly learning multiple aspect embedding vectors for each node is superior to learning an embedding vector for every possible node pairs. **2)** Recall that there are multiple types of nodes in a HetNet, and some types of nodes are more helpful than others in determining the aspect of a target node. Table 6 shows the performance when different types of context nodes are used for determining the aspect of the target node, i.e., author node. We observe that using paper (type) only performs the best, whereas using author only performs the worst. This is expected because we encode the paper abstract to generate paper embeddings, and thus the aspect of the target author can be readily determined by examining paper nodes within its context nodes. On the other hand, the author embeddings only capture the local structural information, which is less deterministic of the target aspect compared to the node attributes.

Table 6: Using different types of context for asp2vec-het.

asp2vec-het				
dim= 200 ($d = 40$, $K = 5$)		Recall@10	F1@10	AUC
Context Type	Author	0.7627	0.2677	0.9099
	Paper	**0.7701**	**0.2709**	**0.9137**
	Author+Paper	0.7682	0.2695	0.9127

6 CONCLUSION

In this paper, we present a novel multi-aspect network embedding method called asp2vec that dynamically determines the aspect based on the context information. Major components of asp2vec is 1) the aspect selection module, which is based on the Gumbel-Softmax trick to approximate the discrete sampling of the aspects, and facilitate end-to-end training of the model, and 2) the aspect regularization framework that encourages the learned aspect embeddings to be diverse, but to some extent related to each other. We also demonstrate how to extend asp2vec to a HetNet. Through experiments on multiple networks with various types, we empirically show the superiority of our proposed framework.

REFERENCES

[1] Sami Abu-El-Haija, Bryan Perozzi, and Rami Al-Rfou. 2017. Learning edge representations via low-rank asymmetric projections. In *CIKM*.
[2] Babajide O Ayinde, Tamer Inanc, and Jacek M Zurada. 2019. Regularizing deep neural networks by enhancing diversity in feature extraction. *TNNLS* (2019).
[3] Hongxu Chen, Hongzhi Yin, Weiqing Wang, Hao Wang, Quoc Viet Hung Nguyen, and Xue Li. 2018. PME: projected metric embedding on heterogeneous networks for link prediction. In *KDD*.
[4] Ting Chen and Yizhou Sun. 2017. Task-guided and path-augmented heterogeneous network embedding for author identification. In *WSDM*.
[5] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *KDD*.
[6] Alessandro Epasto and Bryan Perozzi. 2019. Is a single embedding enough? learning node representations that capture multiple social contexts. In *WWW*.
[7] Tao-yang Fu, Wang-Chien Lee, and Zhen Lei. 2017. Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning. In *CIKM*.
[8] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *KDD*.
[9] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *NeurIPS*.
[10] Binbin Hu, Chuan Shi, Wayne Xin Zhao, and Philip S Yu. 2018. Leveraging meta-path based context for top-n recommendation with a neural co-attention model. In *KDD*.
[11] Rana Hussein, Dingqi Yang, and Philippe Cudré-Mauroux. 2018. Are Meta-Paths Necessary? Revisiting Heterogeneous Graph Embeddings. In *CIKM*.
[12] Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144* (2016).
[13] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).
[14] Qimai Li, Zhichao Han, and Xiao-Ming Wu. 2018. Deeper insights into graph convolutional networks for semi-supervised learning. In *AAAI*.
[15] Ninghao Liu, Qiaoyu Tan, Yuening Li, Hongxia Yang, Jingren Zhou, and Xia Hu. 2019. Is a single vector enough? exploring node polysemy for network embedding. In *KDD*.
[16] Jianxin Ma, Peng Cui, Kun Kuang, Xin Wang, and Wenwu Zhu. 2019. Disentangled graph convolutional networks. In *ICML*.
[17] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NeurIPS*.
[18] Chanyoung Park, Donghyun Kim, Jiawei Han, and Hwanjo Yu. 2019. Unsupervised Attributed Multiplex Network Embedding. *arXiv preprint arXiv:1911.06750* (2019).
[19] Chanyoung Park, Donghyun Kim, Qi Zhu, Jiawei Han, and Hwanjo Yu. 2019. Task-Guided Pair Embedding in Heterogeneous Network. In *CIKM*.
[20] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *KDD*.
[21] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618* (2012).
[22] Leonardo FR Ribeiro, Pedro HP Saverese, and Daniel R Figueiredo. 2017. struc2vec: Learning node representations from structural identity. In *KDD*.
[23] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *ESWC*.
[24] Yu Shi, Qi Zhu, Fang Guo, Chao Zhang, and Jiawei Han. 2018. Easing embedding learning by comprehensive transcription of heterogeneous information networks. In *KDD*.
[25] Fan-Yun Sun, Meng Qu, Jordan Hoffmann, Chin-Wei Huang, and Jian Tang. 2019. vGraph: A Generative Model for Joint Community Detection and Node Representation Learning. In *NeurIPS*.
[26] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).
[27] Petar Veličković, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2018. Deep graph infomax. *arXiv preprint arXiv:1809.10341* (2018).
[28] Hao Wang, Tong Xu, Qi Liu, Defu Lian, Enhong Chen, Dongfang Du, Han Wu, and Wen Su. 2019. MCNE: An End-to-End Framework for Learning Multiple Conditional Network Representations of Social Network. In *KDD*.
[29] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous graph attention network. In *WWW*.
[30] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S Yu. 2019. A comprehensive survey on graph neural networks. *arXiv preprint arXiv:1901.00596* (2019).
[31] Liang Yang, Yuanfang Guo, and Xiaochun Cao. 2018. Multi-facet network embedding: Beyond the general solution of detection and representation. In *AAAI*.
[32] Chuxu Zhang, Chao Huang, Lu Yu, Xiangliang Zhang, and Nitesh V Chawla. 2018. Camel: Content-aware and meta-path augmented metric learning for author identification. In *WWW*.
[33] Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *ACL*.

APPENDIX FOR REPRODUCIBILITY

A   EXPERIMENTAL DETAILS

A.1   Code and Datasets

asp2vec is implemented using PyTorch. The source code and instructions on how to run the code can be found here[1]. Table 7 shows the urls of the author implementations of the compared methods, and Table 8 shows the urls from which the datasets can be downloaded. Nodes in social networks refer to users, nodes in PPI network refer to protein, nodes Wikipedia refer to words, nodes in Cora (citation network) refer to papers, and nodes in ca-HepTh, ca-AstroPh and 4area (all three are co-authorship networks) refer to authors. Each node in Cora is associated with a bag-of-words representation of a document (size=1433), whereas the remaining datasets do not have node attribute information. Finally, as DGI uses node attribute information, we used the adjacency matrix to initialize the node attribute matrix to apply DGI on datasets without node attribute information (i.e., all but Cora).

Table 7: URL links to the code.

Methods	URL link to the code
DGI	https://github.com/PetarV-/DGI
PolyDW	github.com/ninghaohello/Polysemous-Network-Embedding
Splitter	github.com/google-research/ google-research/tree/master/graph_embedding/persona
camel	https://github.com/chuxuzhang/WWW2018_Camel
TaPEm	https://github.com/pcy1302/TapEM

Table 8: URL links to the datasets.

	URL link to the dataset
Filmtrust	https://www.librec.net/datasets.html#filmtrust
Cora	https://github.com/tkipf/pygcn/tree/master/data/cora
PPI	https://snap.stanford.edu/node2vec/
Wikipedia	https://snap.stanford.edu/node2vec/
Wiki-vote	https://snap.stanford.edu/data/index.html
CiaoDVD	https://www.librec.net/datasets.html#filmtrust
ca-HepTh	https://snap.stanford.edu/data/index.html
BlogCatalog	http://socialcomputing.asu.edu/datasets/BlogCatalog3
ca-AstroPh	https://snap.stanford.edu/data/index.html
4area	http://www.perozzi.net/projects/focused-clustering/
Epinions	http://www.trustlet.org/downloaded_epinions.html
Flickr	http://socialcomputing.asu.edu/datasets/Flickr
DBLP	https://github.com/pcy1302/TapEM/tree/master/data

A.2   Evaluation Protocol

Homogeneous Network: Link Prediction. For link prediction in homogeneous networks, we follow the convention of [1, 6, 8]. More precisely, we first split the original graph into to two equal sized set of edges, i.e., E_{train}, and E_{test}. To obtain E_{test}, we randomly remove the edges while preserving the connectivity of the original graph. To obtain the negative counterparts of E_{train} and E_{test}, we

randomly generate a set of edges of size $2 \times |E_{test}|$ and split it in half into $E_{train_{neg}}$ and $E_{test_{neg}}$, respectively. We train a logistic regression classifier on the $E_{train} \cup E_{train_{neg}}$ [1, 6], and the link prediction performance is then measured by ranking (AUC-ROC) the removed edges E_{test} among $E_{test} \cup E_{test_{neg}}$. Finally, for the above preprocessing, we used [2], which is from the implementation of [1].

Heterogeneous Network: Author Identification. For author identification in a heterogeneous network, we follow the convention of [19, 32]. More precisely, we use papers published before timestamp T (= 2013) for training, and split papers that are published after timestamp T in half to make validation and test datasets. We report the test performance when the validation result is the best. For final evavaluation, we randomly sample a set of negative authors and combine it with the set of true authors to generate 100 candidate authors for each paper, and then rank the positive paper among them.

Table 9: Link prediction result over various d and K under a limited $d \times K$.

	dim= 200 ($d \times K = 200$)			
	$d = 20,$ $K = 10$	$d = 50,$ $K = 4$	$d = 40,$ $K = 5$	$d = 100,$ $K = 2$
Filmtrust	0.7426	0.7455	0.7425	0.7537
Wiki-vote	0.6478	0.6450	0.6480	0.6476
CiaoDVD	0.7430	0.7445	0.7385	0.7370
BlogCatalog	0.9503	0.9538	0.9515	0.9292
Epinions	0.7416	0.7436	0.7441	0.7370
Flickr	0.9584	0.9524	0.9571	0.9629
PPI	0.8887	0.8947	0.8947	0.8530
Wikipedia	0.9049	0.9040	0.8987	0.8783
Cora	0.8814	0.8827	0.8770	0.8701
ca-HepTh	0.8989	0.9051	0.9088	0.9082
ca-AstroPh	0.9734	0.9803	0.9821	0.9812
4area	0.9503	0.9547	0.9573	0.9600

A.3   Experimental Settings

- For all the compared methods, we use the hyperparameters that are reported to be the best by the original papers. For PolyDW [15], number of random walks for each node is 110, the length of each walk is 11, window size is 4, number of samples per walk (R in [15]) is 10, and the number of negative samples is 5. For Splitter [6], number of random walks for each node is 40, the length of each walk is 10, the window size is set to 5, and the number of negative samples is 5. We found that the above setting works the best.

- Prior to training asp2vec, we perform warm-up step. More precisely, we initialize target and aspect embedding vectors of asp2vec with the final trained embedding vecotrs of Deepwalk [20]. We found that the warm-up step usually gives better performance.

- For DW and asp2vec, we follow the setting of node2vec [8] where the number of random walks for each node is 10, the length of each walk is 80, the window size is 3, and the number of negative samples is 2. τ and λ of asp2vec are set to 0.5, and 0.01, respectively. Note that we also tried other settings than those mentioned above, but no significant different results were obtained.

- Moreover, we first tried gensim implementation[3] of DW [20] / node2vec [8] but we found that the results were inconsistent.

---

[1]https://github.com/pcy1302/asp2vec

[2]https://github.com/google/asymproj_edge_dnn/blob/master/create_dataset_arrays.py
[3]https://radimrehurek.com/gensim/models/word2vec.html

Table 10: Link prediction result of asp2vec for various number of aspects given a fixed d.

Num.	dim = $d \times K$ ($d = 20$)								
Aspects (K)	2	3	4	5	6	7	8	9	10
Filmtrust	0.7427	0.7408	0.7381	0.7426	0.7412	0.7398	0.7436	0.7417	0.7430
Wiki-vote	0.6466	0.6474	0.6484	0.6478	0.6479	0.6476	0.6483	0.6483	0.6483
CiaoDVD	0.7419	0.7428	0.7427	0.7430	0.7431	0.7436	0.7435	0.7437	0.7439
BlogCatalog	0.9227	0.9448	0.9504	0.9503	0.9500	0.9489	0.9483	0.9479	0.9475
Epinions	0.7285	0.7389	0.7410	0.7416	0.7415	0.7414	0.7416	0.7415	0.7415
Flickr	0.9267	0.9459	0.9520	0.9584	0.9519	0.9508	0.9492	0.9480	0.9476
PPI	0.8531	0.8814	0.8875	0.8887	0.8891	0.8883	0.8893	0.8896	0.8897
Wikipedia	0.8928	0.8792	0.9022	0.9049	0.9053	0.9045	0.9034	0.9024	0.9009
Cora	0.8831	0.8835	0.8872	0.8814	0.8877	0.8880	0.8869	0.8860	0.8849
ca-HepTh	0.8977	0.9005	0.9001	0.8989	0.8978	0.8962	0.8950	0.8952	0.8955
ca-AstroPh	0.9610	0.9699	0.9730	0.9734	0.9731	0.9728	0.9725	0.9721	0.9717
4area	0.9496	0.9512	0.9511	0.9503	0.9491	0.9479	0.9453	0.9447	0.9438

Algorithm 1: Pseudocode for training asp2vec.

Input : A network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, target embedding matrix $\mathbf{P} \in \mathbb{R}^{|V| \times d}$, aspect embedding matrix $\{\mathbf{Q}_*^{(s)} \in \mathbb{R}^{|V| \times d}\}_{s=1}^K$

Output: Final embedding matrix $\mathbf{U} \in \mathbb{R}^{|V| \times d}$

1 Perform a set of random walks $\mathcal{W}$ on $\mathcal{G}$
2 **while** *Convergence* **do**
3   **foreach** $\mathbf{w} \in \mathcal{W}$ **do**
4      $\mathcal{J}_{\text{asp2vec}}^{(\mathbf{w})} \leftarrow \text{run\_asp2vec}(\mathbf{w})$
5   **end**
6    $\mathcal{L}_{\text{asp2vec}} = -\sum_{\mathbf{w} \in \mathcal{W}} \mathcal{J}_{\text{asp2vec}}^{(\mathbf{w})}$;   // Eqn. 8
7    $\text{reg}_{\text{asp}} = \sum_{i=1}^{K-1} \sum_{j=i+1}^{K} \text{A-Sim}(\mathbf{Q}_*^{(i)}, \mathbf{Q}_*^{(j)})$;   // Eqn. 10
8    $\mathcal{L} = \mathcal{L}_{\text{asp2vec}} + \lambda \, \text{reg}_{\text{asp}}$;   // Eqn. 15
9   Update embedding matrices $\mathbf{P}$ and $\{\mathbf{Q}_*^{(s)}\}_{s=1}^K$ by minimizing $\mathcal{L}$
10 **end**
11 Compute final embedding matrix $\mathbf{U}$;   // Eqn. 16
12 **Function** run_asp2vec($\mathbf{w}$):
13   **foreach** $v_i \in \mathbf{w}$ **do**
14   **foreach** $v_j \in \mathcal{N}(v_i)$ **do**
15   **for** $s = 1; \ s < K; \ s = s + 1$ **do**
16   Compute $p(\delta(v_i) = s | \mathcal{N}(v_i))$ // Eqn. 7
17          $p(v_j | v_i, p(\delta(v_i) = s)) \leftarrow \frac{\exp(\langle \mathbf{P}_i, \mathbf{Q}_j^{(s)} \rangle)}{\sum_{v_{j'} \in \mathcal{V}} \exp(\langle \mathbf{P}_i, \mathbf{Q}_{j'}^{(s)} \rangle)}$
18   **end**
19        $\log p(v_j | v_i) \leftarrow \sum_{s=1}^K p(\delta(v_i) = s | \mathcal{N}(v_i)) \log p(v_j | v_i, p(\delta(v_i) = s))$
20   **end**
21      $\mathcal{J}_{\text{asp2vec}}^{(\mathbf{w}), \mathcal{N}(v_i)} \leftarrow \sum_{v_j \in \mathcal{N}(v_i)} \log p(v_j | v_i)$
22   **end**
23    $\mathcal{J}_{\text{asp2vec}}^{(\mathbf{w})} \leftarrow \sum_{v_i \in \mathbf{w}} \mathcal{J}_{\text{asp2vec}}^{(\mathbf{w}), \mathcal{N}(v_i)}$ // Eqn. 2
24   **return** $\mathcal{J}_{\text{asp2vec}}^{(\mathbf{w})}$

Specifically, for BlogCatalog, PPI, Wikipedia, and Cora, we found the performance of gensim DW significantly underperformed our implementation of DW. Therefore, we reported the results of our DW/node2vec which produced consistent results that were consistently better than gensim implementation.

- It is important to note that while we successfully reproduced the results of Splitter on ca-HepTh, and ca-AstroPh reported in [6], Wiki-vote could not be reproduced. This is because the implementation of Splitter is intended for an undirected graph, whereas Wiki-vote is a directed graph. We conjecture that Epasto *et al.* [6] converted Wiki-vote into an undirected graph, and reported the results. However, we used a directed version of Wiki-vote as input, which decreased the link prediction performance.
- For fair comparisons, we fixed the number of dimensions for each node throughout all the compared methods. For example, if $d = 100$ for DW, then we set $d = 20$ and $K = 5$ for asp2vec to make the total dimension size 100, even though we do not concatenate the aspect embedding vectors.
- For asp2vec-het, we set the window size to 3, and number of negative samples to 2, and for a fair comparison, we follow [19] and we use a metapath "APA".

Other Results. Table 10 shows the results for various number of aspects K when d is fixed to 20. We observe that $K = 5$ consistently gives competitive results albeit not the best for some cases. This is the reason why we fixed $K = 5$ in our experiments. Table 9 shows results for different combinations of d and K that multiplied to be 200. We observe that the best combination of d and K differs according to the datasets, where we could observe even better results compared with the results reported in our paper. Again, since $d = 50$ and $K = 4$ consistently gives competitive results, we used it for our experiments with dim=200 in our paper. Algorithm 1 shows the pseudocode for training asp2vec.