Task-Guided Pair Embedding in Heterogeneous Network

Chanyoung Park[1], Donghyun Kim[2], Qi Zhu[1], Jiawei Han[1], Hwanjo Yu[3]*
[1]University of Illinois at Urbana-Champaign, IL, USA, [2]Yahoo! Research, CA, USA
[3]Pohang University of Science and Technology (POSTECH), Pohang, South Korea
{pcy1302,qiz3,hanj}@illinois.edu,donghyun.kim@verizonmedia.com,hwanjoyu@postech.ac.kr

ABSTRACT

Many real-world tasks solved by heterogeneous network embedding methods can be cast as modeling the likelihood of a pairwise relationship between two nodes. For example, the goal of author identification task is to model the likelihood of a paper being written by an author (paper–author pairwise relationship). Existing task-guided embedding methods are node-centric in that they simply measure the similarity between the node embeddings to compute the likelihood of a pairwise relationship between two nodes. However, we claim that for task-guided embeddings, it is crucial to focus on *directly* modeling the pairwise relationship. In this paper, we propose a novel task-guided pair embedding framework in heterogeneous network, called TaPEm, that directly models the relationship between a pair of nodes that are related to a specific task (e.g., paper-author relationship in author identification). To this end, we 1) propose to learn a pair embedding under the guidance of its associated context path, i.e., a sequence of nodes between the pair, and 2) devise the pair validity classifier to distinguish whether the pair is valid with respect to the specific task at hand. By introducing pair embeddings that capture the semantics behind the pairwise relationships, we are able to learn the fine-grained pairwise relationship between two nodes, which is paramount for task-guided embedding methods. Extensive experiments on author identification task demonstrate that TaPEm outperforms the state-of-the-art methods, especially for authors with few publication records.

KEYWORDS

Heterogeneous Network, Author Identification, Representation Learning, Deep Learning

ACM Reference format:
Chanyoung Park[1], Donghyun Kim[2], Qi Zhu[1], Jiawei Han[1], Hwanjo Yu[3]*. 2019. Task-Guided Pair Embedding in Heterogeneous Network. In *Proceedings of The 28th ACM International Conference on Information and Knowledge Management, Beijing, China, November 3–7, 2019 (CIKM '19),* 10 pages.
https://doi.org/10.1145/3357384.3357982

1 INTRODUCTION

The goal of network embedding is to learn low dimensional representations for nodes in a network while preserving the network structure [12, 23] and properties [11, 14, 22, 33, 36]. As a large number of social and information networks are heterogeneous in nature, i.e., nodes and edges are of multiple types, heterogeneous network embedding methods have recently garnered attention [29–31, 35]. They learn node embeddings by exploiting various types of relationships among nodes and the network structure, and use them for general downstream tasks, such as node classification [7, 26], link prediction [9, 27], and clustering [3, 15].

Recent studies have shown that previous heterogeneous network embedding methods fall short of solving a specific task, such as anomaly detection [5], recommendation [25], sentiment link prediction [32], and author identification [4, 34], to name a few, because they learn general purpose node embeddings. In this regard, recently proposed task-guided network embedding methods guide node embeddings to perform well on a specific task by introducing a task-specific objective, instead of learning general purpose node embeddings that preserve the overall proximity among nodes. For example, Shi et al., generate node sequences that are meaningful for recommendation by meta-path based random walk [7], and integrate them with matrix factorization to specifically optimize for the rating prediction task [25]. Moreover, Wang et al., integrate users' sentiment relation network, social relation network and profile knowledge network into a heterogeneous network to specifically optimize for the sentiment link prediction task [32].

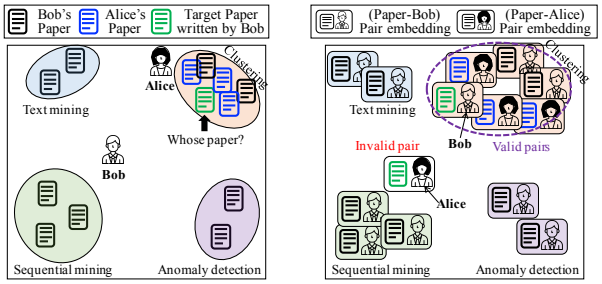
However, for task-guided embedding methods, it is crucial to particularly focus on *directly modeling the pairwise relationship between two nodes*, because their ultimate goal is usually to model the likelihood of the pairwise relationship. i.e., the link probability between two nodes. For example, for recommendation, the goal is to model the likelihood of a user favoring an item (i.e., user–item pairwise relationship). For author identification, the goal is to model the likelihood of a paper being written by an author (i.e., paper–author pairwise relationship). Nevertheless, previous task-guided embedding methods are node-centric in that they learn task-guided *node embeddings*, and then simply compute the likelihood of a pairwise relationship between two nodes by employing a similarity metric, such as inner product [5, 25, 32] or Euclidean distance [34], between the learned node embeddings.

In the light of this issue, we propose a novel **Ta**sk-guided **P**air **Em**bedding framework in heterogeneous network, called TaPEm, that directly embeds a pair of nodes whose likelihood we wish to model. i.e., task-guided *pair embedding*. Unlike previous node-centric task-guided embedding methods, the task-specific objective of TaPEm is derived from the pair embedding. Our intuition is that if a pair embedding can capture the semantics behind the relationship between two nodes that constitute the pair, we can model the fine-grained pairwise relationship better than the case when each node only has a single embedding. As an illustration, consider the following toy example.

*Corresponding Author.

CIKM '19, November 3–7, 2019, Beijing, China

ACM ISBN 978-1-4503-6976-3/19/11…$15.00
https://doi.org/10.1145/3357384.3357982



(a) Node embedding (b) Pair embedding

Figure 1: Comparisons of example visualization between node embedding and pair embedding.

Toy Example. Take an example of author identification scenario, which aims to find the authors of anonymous papers [4, 34]. Figure 1(a) illustrates an example visualization of embeddings of papers and authors, where each author is assigned a single embedding. We assume that Bob has written multiple papers in various research areas, whereas Alice's work are solely devoted to "clustering"; hence, Alice is embedded close to papers whose topics are "clustering". Since each author has a single embedding, it has to be embedded to a single point that is optimal considering all of his diverse research areas. However, a problem arises if we were to identify a true author of "Target Paper" about "clustering" (in green), which is written by Bob. In this case, since Alice, who has written papers only on "clustering", is likely to be embedded closer to papers on "clustering" than Bob, Bob will be eventually ranked lower than Alice, which is not a desired result. On the other hand, as shown in Figure 1(b), if we can embed each paper–author pair such that each pair embedding independently captures its associated research topic and its validity information, ("Target paper", Bob) pair can be embedded closer to the valid pairs related to "clustering" than invalid ("Target paper", Alice) pair is.

In this regard, the key for successfully learning a task-guided pair embedding boils down to modeling 1) the semantics (e.g., research topic) behind the pairwise relationship, and 2) the validity of the pair regarding a specific task (e.g., given a paper–author pair, whether the paper in the pair is written by the author in the pair). Note that we will continue our discussions on the author identification scenario hereafter for the ease of explanation.

To capture the semantics behind the pairwise relationship, we propose to explicitly encode the paths between a paper–author pair, where we denote such paths as *context paths*. More precisely, from meta-path guided random walks [7], we first extract paper–author pairs each of which is associated with multiple context paths (**Sec. 4**). Then, we embed each pair (**Sec. 5.1.1**) and its associated context path (**Sec. 5.1.2**) into a vector, respectively. Under the assumption that a context path reveals the research topic associated with the pair, we make the pair embedding naturally get similar to the embeddings of more frequently appearing context paths (**Sec. 5.1.3**). By encoding the related research topic into a paper–author pair embedding, we can model the fine-grained pairwise relationship between the paper–author pair. In the meantime, for each paper–author pair, we introduce a pair validity classifier to distinguish whether the pair is valid or not with respect to the specific task at hand (**Sec. 5.2**). By reflecting the pair validity information into the pair embedding, we make two nodes that constitute a pair close to each other not only if they are related to a similar research topic, but also the pair itself is valid at the same time. The pair embeddings obtained from the above process eventually makes it easier to distinguish the valid pairs from the invalid ones.

To verify the benefit of our task-guided pair embedding framework in heterogeneous network, we specifically focus on the task of **author identification** in big scholarly data [24]. Our extensive experiments demonstrate that TaPEm considerably outperforms the state-of-the-art task-guided heterogeneous embedding methods, especially for authors with few publication records (**Sec. 6.2**). We also perform various experiments to qualitatively ascertain the benefit of the pair embedding framework of TaPEm.

2 RELATED WORK

Network Embedding. The goal of network embedding is to learn a low dimensional representation for each node of a network while preserving the network structure and various properties such as attributes related to nodes [3, 13, 14] and edges [11]. The learned embeddings are then used for general downstream tasks, such as node classification [1], link prediction [22, 36], and clustering [33]. Inspired by the recent advancement of word embedding techniques in natural language processing [21], numerous network embedding approaches based on random walk have been proposed [12, 23]. DeepWalk [23] and node2vec [12] combine random walk and skip-gram to learn node embeddings. However, as these methods are proposed for homogeneous networks in which nodes and edges are of a single type, and thus are not suitable for modeling heterogeneous networks, there has been a line of research on heterogeneous network embedding [7, 9, 27, 32, 35]. Specifically, metapath2vec [7] proposed a random walk scheme that is conditioned on meta-paths, and learned node embeddings by heterogeneous skip-gram with negative sampling. JUST [15] tackled the limitation of meta-path based random walk, and proposed random walks with jump and stay strategies. Hin2Vec [9] considered the relationship between nodes to learn node embeddings, but it considered all possible relationships between two nodes aiming at learning general node embeddings, which is distinguished from our proposed framework in that we specifically focus on a single relationship related to a specific task. Although these methods have been shown to be effective on general downstream tasks such as node classification, clustering and similarity search, they are not specifically designed to perform well on specific tasks such as recommendation [25], anomaly detection [5], sentiment link prediction [32] and author identification [4, 34]. We propose a novel task-guided pair embedding framework in heterogeneous network, and focus on the problem of author identification as an application of our framework.

Author Identification. Many conferences in computer science adopt the double-blind review policy to eliminate bias in favor of well-known authors. However, a knowledgeable reviewer can often disclose the authors of a paper by its content [4, 34]. In this regard, the effectiveness of the double-blind review process is constantly being questioned by the research community [2, 10].

To specifically focus on the author identification task, Chen and Sun proposed a task-guided heterogeneous network embedding

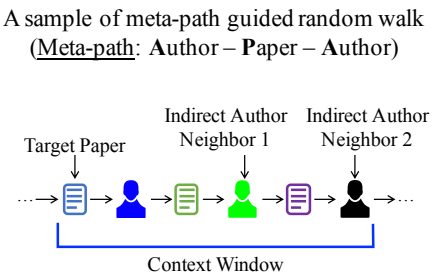


Figure 2: Example showing the behavior of skip-gram model of Camel on a sample meta-path guided random walk.

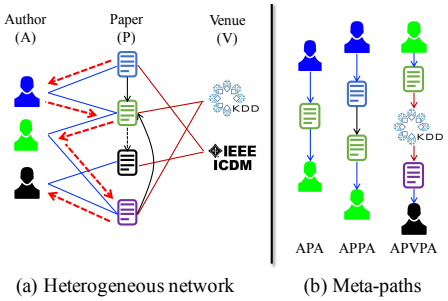


Figure 3: An illustrative example of (a) academic heterogeneous network. (b) Different meta-path schemes.

method, called HNE, that learns embeddings for authors, and different types of information of a paper, such as keywords, references and venues [4]. More recently, Zhang et al., proposed Camel [34] that encodes the semantic content of papers (i.e., abstract) instead of keywords, and considers indirectly correlated paper–author pairs obtained from meta-path guided random walk on the academic heterogeneous network using heterogeneous skip-gram model [7]. Although Camel has shown its effectiveness, it suffers from an inherent limitation that each author has a single embedding even though authors may have diverse research areas (Refer to **Toy Example** in Figure 1(a)). Another limitation of Camel is that it inadvertently makes a paper embedding and an author embedding similar to each other if they frequently appear together within a context window of a random walk sequence, whether or not the author is a true author of the paper. Figure 2 shows a sample segment of a meta-path guided random walk. The skip-gram model of Camel trains the embedding of "Target Paper" to be similar to the embeddings of both Indirect Author Neighbor 1 and 2, regardless of their true authorship. i.e., whether or not they are the true authors of "Target Paper". After performing multiple meta-path guided random walks, it is natural that an active author (i.e., an author with many publications) is more likely to appear frequently together with "Target paper" within the same context window than an inactive author. Therefore, according to the above skip-gram model, if an active author is a true author of "Target paper", then Camel can provide correct predictions. However, if an inactive author is a true author of "Target paper", Camel performs poorly because the inactive author does not appear frequently together with "Target paper", and thus not trained enough to be similar to "Target paper". To make the matter worse, if an active author is not a true author but appears frequently with "Target paper", Camel will still predict the active author as the true author. In other words, *Camel is biased to active authors*, which is a consequence of the skip-gram model. While such behavior derived from the skip-gram model may be rational for general network embedding tasks whose goal is to preserve the overall proximity between nodes [7, 9, 12, 23], the skip-gram based objective should be reconsidered when it comes to a specific task, such as author identification. We later show in our experiments that TaPEm is robust to the activeness of the authors, thanks to the pair validity classifier.

There also exist several recent work that learn relation type-specific node embeddings [26, 27]. However, the relation labels between nodes in their work are given in advance (e.g., a connection between two movies implies which genre they share), whereas relations between papers and authors in academic networks do not have predefined edge labels; we only know that a link exists between a paper and an author, if the paper is written by the author. Therefore, these methods cannot be directly applied to our setting.

3 PROBLEM DEFINITION

In this section, we first introduce preliminary concepts regarding heterogeneous network, and then formalize the task to be addressed.

Definition 3.1. (**Heterogeneous Network**) A heterogeneous network is a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{T}_v, \mathcal{T}_e, \phi, \psi)$ in which $\mathcal{V}$ is the union of different types of nodes, $\mathcal{E}$ is the union of different types of edges. Each node $v \in \mathcal{V}$ and edge $e \in \mathcal{E}$ are associated with a node type mapping function $\phi : \mathcal{V} \rightarrow \mathcal{T}_v$, and an edge type mapping function $\psi : \mathcal{E} \rightarrow \mathcal{T}_e$, respectively. $\mathcal{T}_v$ and $\mathcal{T}_e$ denote the sets of node types and edge types, respectively. $\mathcal{G}$ is defined as a heterogeneous network when the number of node types $|\mathcal{T}_v| > 1$ or the number of edge types $|\mathcal{T}_e| > 1$.

Example. Figure 3(a) shows an academic heterogeneous network that consists of three different types of nodes (Author (A), Paper (P), and Venue (V)), and three different types of edges (A↔P: author writes paper, P→P: paper cites paper, and P↔V: paper publishes in venue).

Definition 3.2. (**Meta-path**) Given $A_i \in \mathcal{T}_v$ and $R_i \in \mathcal{T}_e$, a meta-path [30] $\mathcal{P}$ is defined as a path in the form of $A_1 \xrightarrow{R_1} A_2 \xrightarrow{R_2} \cdots \xrightarrow{R_{l-1}} A_l$ (abbreviated as $A_1A_2 \cdots A_l$), which describes a composite relation $R = R_1 \circ R_2 \circ \cdots \circ R_{l-1}$ between objects A_1 and A_l, where $\circ$ denotes the composition operator on relations.

Example. Figure 3(b) shows three different ways that two authors can be connected: Author-Paper-Author (APA), Author-Paper-Paper-Author (APPA), and Author-Paper-Venue-Paper-Author (APVPA). Each meta-path contains different semantics. Specifically, APA means co-authorship, APPA means an author cites a paper written by another author, and APVPA means two authors publishing papers in the same venue.

Definition 3.3. (**Meta-path guided Random Walk**) A meta-path guided random walk [7] $\boldsymbol{w} \in \mathcal{W}^{\mathcal{P}}$ is a random walk guided by a specific meta-path $\mathcal{P}$, where $\mathcal{W}^{\mathcal{P}}$ is a set of collected random walks. Each random walk recursively samples a specific $\mathcal{P}$ until the walk length reaches a predefined value.

Example. Red dashed arrows in Figure 3(a) shows a segment of a sample random walk guided by meta-path APA. In other words, a random walker in each step should only follow the pattern APA when deciding the next step.

Definition 3.4. (**Context path**) Given two nodes $v_i, v_j \in \mathcal{V}$, a set of context paths from v_i to v_j for all walks $\boldsymbol{w} \in \mathcal{W}^{\mathcal{P}}$ are denoted by $C^{\mathcal{P}}_{i \to j}$.

Example. Figure 4 shows an unfolded view of the sample random walk that follow the red dashed line shown in Figure 3(a). Each red box denotes a context path of the associated paper–author pair in dashed circles.

The task to be addressed is formally defined as follows:

Given: A set of node pairs (v_i, v_j) and their associated set of context paths $C^{\mathcal{P}}_{i \to j}$ extracted from multiple random walks guided by a meta-path $\mathcal{P}$ in a set of meta-path scheme $\mathcal{S}(\mathcal{P})$,
Goal: Predict the likelihood of the pairwise relationship between any two nodes in $\mathcal{V}$.

4 HETEROGENEOUS NETWORK EMBEDDING

Inspired by skip-gram based word2vec [20, 21], previous network representation learning methods [12, 23] viewed a network as a document. These methods first perform random walks on a network to extract multiple sequences of nodes, which are analogous to sequences of words, and then apply the skip-gram model to learn the representation of a node. However, these methods only focus on homogeneous networks in which nodes and edges are of a single type. To learn effective representations of nodes in a heterogeneous network, metapath2vec [7] introduced the meta-path guided random walk scheme together with the heterogeneous skip-gram model.

Meta-path Guided Random Walk. Given a heterogeneous network $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{T}_v, \mathcal{T}_e, \phi, \psi)$ and a meta-path $\mathcal{P} : A_1 \xrightarrow{R_1} A_2 \xrightarrow{R_2} \cdots A_i \xrightarrow{R_i} A_{i+1} \cdots \xrightarrow{R_{l-1}} A_l$, it first performs meta-path guided random walks to generate paths that capture both the semantic and structural correlations between multiple types of nodes. The transition probability of walk at step t is defined as:

$$p(v^{t+1}|v_i^t, \mathcal{P}) = \begin{cases} \frac{1}{|N_{i+1}(v_i^t)|}, & (v^{t+1}, v_i^t) \in \mathcal{E}, \psi(v^{t+1}) = i+1 \\ 0, & (v^{t+1}, v_i^t) \in \mathcal{E}, \psi(v^{t+1}) \neq i+1 \\ 0, & (v^{t+1}, v_i^t) \notin \mathcal{E} \end{cases} \tag{1}$$

where $v_i^t \in A_i$, and $N_{i+1}(v_i^t)$ denotes the A_{i+1} type of neighborhood of node v_i^t. That is, the flow of a meta-path guided random walk is conditioned on the meta-path $\mathcal{P}$, and thus $v^{t+1} \in A_{i+1}$.

Heterogeneous Skip-gram Model. After generating a set of walks $\mathcal{W}^{\mathcal{P}}$ under meta-path $\mathcal{P}$ by performing the meta-path guided random walk as described above, metapath2vec learns the representation of nodes by maximizing the probability of having the heterogeneous context $N_t(v), t \in \mathcal{T}_v$ given a node v:

$$\arg\max_{\theta} \sum_{v \in \mathcal{V}} \sum_{t \in \mathcal{T}_v} \sum_{x_t \in N_t(v)} \log p\left(x_t | v; \theta\right) \tag{2}$$

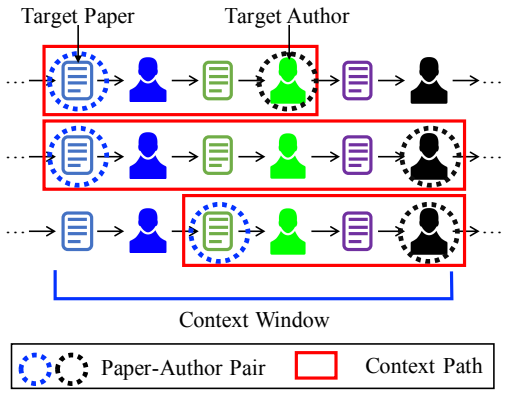


Figure 4: Examples of possible paper–author pairs, and their associated context paths within a segment of a meta-path guided random walk.

where x_t is a t-th type context node of v. The likelihood probability $p(x_t|v;\theta)$ is commonly defined as a softmax function, and Eqn. 2 is optimized by adopting the negative sampling technique [7, 21].

5 THE PAIR EMBEDDING FRAMEWORK

We present our novel task-guided pair embedding framework in heterogeneous network, called TaPEm. As previously stated, we focus on the task of author identification as an application of our framework, because this task can be cast as predicting the likelihood of the pairwise relationship between a paper and an author. We first formally define the task of author identification as follows:

Task: Author Identification. Given a set of papers published before timestamp T, each of which is associated with bibliographic information, such as authors, references, abstract, published year and venue, our task is to rank all potential authors of each paper published after timestamp T, such that top-ranked authors are true authors.

5.1 Context Path-aware Pair Embedding

To begin with, we perform multiple meta-path guided random walks as in Eqn. 1 on the academic heterogeneous network, from which we extract paper–author pairs, and the associated context paths of each pair. Figure 4 shows how paper–author pairs and their associated context paths are extracted from a segment of a sample meta-path guided random walk. More precisely, given a paper–author pair in dashed circles, the associated context path is defined by a sequence of nodes between the paper and the author, including themselves. Note that we can generate three different pair-path instances from the random walk segment shown in Figure 4. In the following subsections, we will explain how the pairs and their context paths are embedded, respectively.

5.1.1 ***Embedding Paper–Author Pair.*** Given a combination of paper embedding $\mathbf{p}_v \in \mathbb{R}^K$ and author embedding $\mathbf{q}_u \in \mathbb{R}^K$ as $\mathbf{Comb}(\mathbf{p}_v, \mathbf{q}_u) \in \mathbb{R}^{4K}$, the pair embedder $\mathbf{g} : \mathbb{R}^{4K} \to \mathbb{R}^d$ is a multi-layer perceptron (MLP) with n layers that generates a d-dimensional embedding for a paper–author pair (v, u).

$$h^{(l)} = \begin{cases} \text{ReLU}(W^{(l)}\text{Dropout}(h^{(l-1)}) + b^{(l)}), & 0 < l < n \\ W^{(l)}\text{Dropout}(h^{(l-1)}) + b^{(l)}, & l = n \end{cases} \quad (3)$$

$$h^{(0)} = \textbf{Comb}(\mathbf{p}_v, \mathbf{q}_u) = [\mathbf{p}_v; \mathbf{q}_u; \mathbf{p}_v \circ \mathbf{q}_u; \mathbf{p}_v - \mathbf{q}_u] \in \mathbb{R}^{4K} \quad (4)$$

$$\mathbf{g}(v, u) = h^{(n)} \in \mathbb{R}^d \quad (5)$$

where $\circ$ denotes element-wise vector multiplication, and $\text{ReLU}(x) = max(0, x)$. We apply dropout [28] on the hidden layers, and take the last layer output of MLP as the pair embedding. In the experiments, $\mathbf{g}(\cdot)$ is a 2-layered MLP each layer with 100 hidden units.

For author identification, we need to represent a paper using information related to the paper. HNE [4] represents a paper by combining the embeddings of its references, keywords, and venue. However, as the semantic content of a paper is critical for identifying the authors of the paper, Camel uses word embeddings [21] to represent a paper by its content [34], i.e., abstract, which considerably outperformed HNE. Hence, we also use the words in the abstract of a paper to represent the paper:

$$\mathbf{p}_v = \text{PaperEncoder}(O_v) \in \mathbb{R}^K \quad (6)$$

where O_v is the index of paper v, and $\text{PaperEncoder}(\cdot)$ is a GRU–based content encoder, which encodes a paper into a vector. We adopt $\mathcal{L}_{Metric}$ of Camel [34] to encode the paper content. Please refer to the original paper for more details [34].

5.1.2 ***Embedding Context Path***. Recall from Figure 4 that given a paper–author pair (v, u) on a meta-path guided random walk guided by $\mathcal{P}$, there exists a set of context paths $C^{\mathcal{P}}_{v\to u}$, i.e., a set of node sequences between paper v and author u. Our assumption is that we can readily *infer the research topic related to the pair* (v, u) *by examining the path between paper* v *and author* u. For example, in the sample random walk shown in Figure 5, P1 and P2 are likely to be about a similar research topic because both of them are written by A1. Besides, A1 and A2 are likely to share a common research interest because they co-authored P2. In short, the context path from P1 to A2 reveals the research topic related to the pair (P1, A2).

Given a set of paper–author pairs and their associated context paths obtained from meta-path guided random walk [7, 34], we embed a context path composed of a sequence of nodes by applying bidirectional gated recurrent unit (GRU) [6], which is commonly used for sequence modeling. More precisely, the context path embedder $\mathbf{f} : \mathbb{R}^K \to \mathbb{R}^d$ is a bidirectional GRU that generates a d-dimensional vector for each context path by adopting an attention module. Take a context path between paper v and author u, i.e., $c \in C^{\mathcal{P}}_{v\to u}$, as an example, which is represented as a sequence of nodes, i.e., $c = \{v, c_2, c_3, ..., c_{n-1}, u\}$. We convert this sequence of nodes into a sequence of K-dimensional embedding vectors $\{\mathbf{p}_v, \mathbf{x}_2, \mathbf{x}_3, ..., \mathbf{x}_{n-1}, \mathbf{q}_u\}$, where $\mathbf{p}_v = \mathbf{x}_1$ and $\mathbf{q}_u = \mathbf{x}_n$. Note that the types of nodes $c_2, c_3..., c_{n-1}$ depend on what kind of meta-path $\mathcal{P}$ guided the random walk. For example, if $\mathcal{P}$ =Author $\to$ Paper $\to$ Author, then $\phi(c_2)$ = Author, $\phi(c_3)$ = Paper, and $\phi(c_{n-1})$ = Paper. At time t, GRU computes the hidden state $\mathbf{h}_t \in \mathbb{R}^d$ given the previous hidden state $\mathbf{h}_{t-1} \in \mathbb{R}^d$ and the current input $\mathbf{x}_t \in \mathbb{R}^K$,

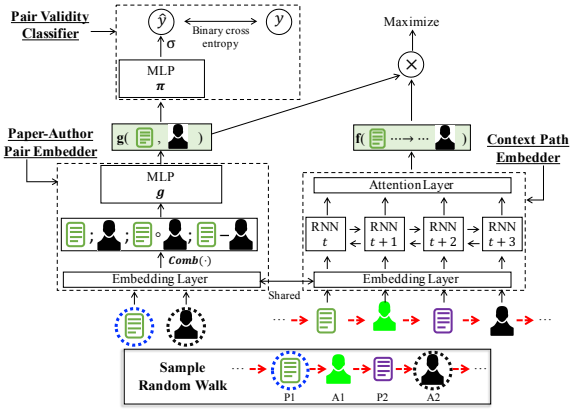


Figure 5: An overview of TaPEm.

i.e., $\mathbf{h}_t = \text{GRU}(\mathbf{h}_{t-1}, \mathbf{x}_t)$. More precisely,

$$\begin{aligned} \mathbf{z}_t &= \sigma(\mathbf{W}_z\mathbf{x}_t + \mathbf{U}_z\mathbf{h}_{t-1} + \mathbf{b}_z), \;\; \mathbf{r}_t = \sigma(\mathbf{W}_r\mathbf{x}_t + \mathbf{U}_r\mathbf{h}_{t-1} + \mathbf{b}_r) \\ \hat{\mathbf{h}}_t &= \tanh[\mathbf{W}_h\mathbf{x}_t + \mathbf{U}_h(\mathbf{r}_t \circ \mathbf{h}_{t-1}) + \mathbf{b}_h] \\ \mathbf{h}_t &= \mathbf{z}_t \circ \mathbf{h}_{t-1} + (1 - \mathbf{z}_t) \circ \hat{\mathbf{h}}_t \end{aligned} \quad (7)$$

where $\sigma(\cdot)$ is a sigmoid function, $\mathbf{W} \in \mathbb{R}^{d\times K}$, $\mathbf{U} \in \mathbb{R}^{d\times d}$ and $\mathbf{b} \in \mathbb{R}^d$ are parameters of GRU network. In order to make full use of the context information from both directions, we apply bidirectional GRU, i.e., $\mathbf{h}_t = \text{BiGRU}(\overrightarrow{\mathbf{h}}_{t-1}, \overleftarrow{\mathbf{h}}_{t-1}, \mathbf{x}_t)$ and combine the output from each direction by a linear projection layer. More precisely,

$$\begin{aligned} \overrightarrow{\mathbf{h}}_t &= \text{GRU}(\overrightarrow{\mathbf{h}}_{t-1}, \mathbf{x}_t), \;\; \overleftarrow{\mathbf{h}}_t = \text{GRU}(\overleftarrow{\mathbf{h}}_{t-1}, \mathbf{x}_t) \\ \mathbf{h}_t &= \mathbf{W}_{\text{proj}}[\overrightarrow{\mathbf{h}}_t; \overleftarrow{\mathbf{h}}_t] + \mathbf{b}_{\text{proj}} \end{aligned} \quad (8)$$

where $\mathbf{W}_{\text{proj}} \in \mathbb{R}^{d\times 2d}$ and $\mathbf{b}_{\text{proj}} \in \mathbb{R}^d$ are parameters of the linear projection layer, and $[\cdot;\cdot]$ denotes vector concatenation operator.

After embedding n nodes within a context path $c \in C^{\mathcal{P}}_{v\to u}$ into an embedding matrix $\mathbf{h} \in \mathbb{R}^{n\times d}$, we aggregate the matrix $\mathbf{h}$ by applying attentive pooling [37] that extracts a d-dimensional vector from $\mathbf{h}$, which summarizes the context path by measuring the contribution of each node in the context path to form a high-level representation of the entire context path. More precisely, the path embedder $\mathbf{f}$ is defined as follows:

$$w_t = \text{softmax}(\mathbf{k}\mathbf{h}_t) = \frac{\exp(\mathbf{k}\mathbf{h}_t)}{\sum_{i=1}^{n}\exp(\mathbf{k}\mathbf{h}_i)}, \;\; \mathbf{f}(c) = \sum_t w_t\mathbf{W}_{\text{attn}}\mathbf{h}_t \quad (9)$$

where $\mathbf{k} \in \mathbb{R}^d$ and $\mathbf{W}_{\text{attn}} \in \mathbb{R}^{d\times d}$. The attention module enables us to pay attention to more important segments in a context path. Note that we have also tried other types of attention [18], but attentive pooling performed the best. We conjecture that as attentive pooling is specifically designed for relation classification tasks [37], it is suitable for author identification in which identifying the paper–author relationship is critical. In short, we expect $\mathbf{f}(c) \in \mathbb{R}^d$ to encode the research topic associated with the context path c.

5.1.3 ***Injecting Context Information into Pairs***. Given a paper–author pair (v, u), our goal is to minimize the negative log likelihood of (v, u) given a context path $c \in C^{\mathcal{P}}_{v\to u}$:

$$\mathcal{L}_{\text{ctx}}(v, u) = \sum_{c\in C^{\mathcal{P}}_{v\to u}} -\log p((v, u)|c, \mathcal{P}) \quad (10)$$

The likelihood probability $p((v,u)|c,\mathcal{P})$ is defined as follows:

$$p((v,u)|c,\mathcal{P}) = \frac{\exp\left[(\mathbf{g}(v,u)\cdot\mathbf{f}(c))\right]}{\sum_{c'\in C_*^{\mathcal{P}}}\exp\left[(\mathbf{g}(v,u)\cdot\mathbf{f}(c'))\right]} \tag{11}$$

where $C_*^{\mathcal{P}}$ denotes all possible context paths that can be obtained from $\mathcal{W}_{\mathcal{P}}$. As directly optimizing Eqn. 11 is computationally expensive owing to a large number of possible context paths $C_*^{\mathcal{P}}$, we apply commonly used negative sampling technique [7, 21, 34] for optimization. More precisely, Eqn. 11 can be approximated as:

$$\log p((v,u)|c,\mathcal{P}) \approx \log\sigma(\mathbf{g}(v,u)\cdot\mathbf{f}(c)) + \sum_{j=1}^{k}\log\sigma\left(-\mathbf{g}(v,u)\cdot\mathbf{f}\left(c^j_{\text{rand}}\right)\right) \tag{12}$$

where k is the number of randomly sampled contexts for pair (v,u). That is, we wish to make $\mathbf{g}(v,u)$ similar to $\mathbf{f}(c)$, if c is a context path between v and u, and keep $\mathbf{g}(v,u)$ dissimilar to the embeddings of random context paths $\mathbf{f}(c_{\text{rand}})$. By maximizing Eqn. 10, the pair embedding $\mathbf{g}(v,u)$ will naturally get similar to the embeddings of more frequently appearing context paths. This in turn facilitates $\mathbf{g}(v,u)$ to encode its related research topic, because frequently appearing paths define the relationship between the pair.

Note that Eqn. 10 has a similar underlying philosophy as the heterogeneous skip-gram model in Eqn. 2, which is to predict context nodes given the center node. However, the difference of our proposed method compared with skip-gram model lies in the perspective that a center node is extended to a pair of nodes, and a context node is extended to a context path.

5.2 Validity of Pair Embedding

Recall that one of the limitations of Camel was that it inadvertently makes a paper embedding and an author embedding similar to each other if they frequently appear together within the same context window of a random walk sequence, whether or not the author is a true author of the paper. Therefore, Camel shows high accuracy when the true authors are active authors with many papers, but performs poorly when the true authors are relatively inactive authors, which is an inevitable consequence of the skip-gram model (refer to our discussions on Figure 2). To this end, we propose a pair validity classifier $\boldsymbol{\pi}:\mathbb{R}^d\rightarrow\mathbb{R}$ to discriminate whether the paper–author pair is a valid pair or not, which is formulated by binary cross-entropy loss as follows:

$$\mathcal{L}_{\text{pv}}(v,u) = y_{v,u}\sigma(\boldsymbol{\pi}(\mathbf{g}(v,u))) + (1-y_{v,u})(1-\sigma(\boldsymbol{\pi}(\mathbf{g}(v,u)))) \tag{13}$$

$$y_{v,u} = \begin{cases} 1, & \text{paper } v \text{ is written by author } u \\ 0, & \text{paper } v \text{ is not written by author } u \end{cases} \tag{14}$$

$\boldsymbol{\pi}(\cdot)$ is a 2-layered MLP with ReLU activation. Armed with the pair validity classifier that explicitly discriminates the validity of each pair with respect to the specific task at hand, TaPEm is able to not only identify active authors with many publications, but also relatively less active authors, because the training of the embedding vectors is no longer solely based on the frequency. Moreover, thanks to the pair validity classifier, two nodes that constitute a pair will be close to each other not only if they are related to a similar research topic, but also the pair itself is valid at the same time.

5.3 Joint Objective

Combining $\mathcal{L}_{\text{ctx}}(v,u)$ and $\mathcal{L}_{\text{pv}}(v,u)$ for all possible (v,u) pairs in meta-path guided random walks guided by $\mathcal{P}$, we obtain the following objective function $\mathcal{L}$:

$$\mathcal{L} = \sum_{\mathcal{P}\in\mathcal{S}(\mathcal{P})}\sum_{\boldsymbol{w}\in\mathcal{W}_{\mathcal{P}}}\sum_{v\in\boldsymbol{w}}\sum_{u\in\boldsymbol{w}[C_v-\tau:C_v+\tau]}\left[\mathcal{L}_{\text{ctx}}(v,u)+\mathcal{L}_{\text{pv}}(v,u)\right] \tag{15}$$

where $\mathcal{S}(\mathcal{P})$ denotes all predefined meta-path schemes, τ is the context window size of paper v, and C_v denotes the position of v in walk w. The final objective function $\mathcal{L}$ can be minimized by using mini-batch Adam optimizer [16]. Figure 5 illustrates the overall model architecture of TaPEm.

Prediction. The final prediction of TaPEm is computed by the output of the pair validity classifier $\boldsymbol{\pi}(\cdot)$. Precisely, if we were to identify true authors of paper v, we rank $\sigma(\boldsymbol{\pi}(\mathbf{g}(v,u)))$ for $u\in U$, where U denotes the set of users, to see how many top-ranked authors are true authors. We only need to know the abstract content of a paper for identifying its true authors, because PaperEncoder$(\cdot)$ and author embeddings are learned during training.

6 EXPERIMENTS

The experiments are designed to answer the following research questions (**RQs**):

- **RQ 1** How does TaPEm perform compared with other state-of-the-art methods?
- **RQ 2** How does TaPEm perform on less active authors (i.e., users with few publications)?
  - Qualitative analysis (**RQ 2-1**)
- **RQ 3** How does each component of TaPEm contribute to the overall performance (Ablation study)?
- **RQ 4** How are the pair embeddings visualized compared with paper/author embeddings?

6.1 Experimental Setup

Dataset. In order to make fair comparisons with Camel [34], we evaluate our proposed method on AMiner dataset[1], which is an academic collaboration platform in computer science domain. As the preprocessed dataset is not available, we preprocess the data to make similar statistics with the dataset used in [34]. More precisely, we extracted 10 years of data from 2006 to 2015, removed the papers published in venues with limited publications (e.g., workshop or tutorial) and papers without abstract text. Moreover, as most researchers pay attention to top venues, and their research areas can be categorized into several different areas, we additionally generate a subset data of six research areas (AMiner-Top) according to Google Scholar Metrics: Artificial Intelligence (AI), Data Mining (DM), Databases (DB), Information System (IS), Computer Vision (CV) and Computational Linguistics (CL). For each research area, we choose three top venues that are considered to have influential papers [2]. In the end, AMiner-Top dataset contains 27,920 authors, 21,808 papers and 18 venues, and AMiner-Full dataset contains 536,811 authors, 447,289 papers and 389 venues.

[1] https://aminer.org/citation

[2] **AI**: ICML, AAAI, IJCAI. **DM**: KDD, WSDM, ICDM. **DB**: SIGMOD, VLDB, ICDE. **IS**: WWW, SIGIR, CIKM. **CV**: CVPR, ICCV, ECCV. **CL**: ACL, EMNLP, NAACL

Table 1: The overall performance on author identification (Impr. denotes improvements of TaPEm over the best baseline).

Dataset		Metric	Sup	MPV	HNE	Camel	$\text{TaPEm}_{\text{npv}}$	TaPEm	Impr.		Sup	MPV	HNE	Camel	$\text{TaPEm}_{\text{npv}}$	TaPEm	Impr.
AMiner-Top	T=2013	Rec@5	0.5460	0.5274	0.4874	0.5902	0.6405	**0.6807**	15.33%	AMiner-All	0.6096	0.5990	0.6110	0.5458	0.7049	**0.7097**	16.15%
		Rec@10	0.6227	0.6746	0.6301	0.7370	0.7677	**0.7849**	6.50%		0.6409	0.7317	0.7166	0.6811	0.8121	**0.8237**	12.57%
		Prec@5	0.2285	0.2148	0.2051	0.2439	0.2662	**0.2835**	16.24%		0.2679	0.2562	0.2679	0.2393	0.3076	**0.3087**	15.23%
		Prec@10	0.1323	0.1401	0.1334	0.1555	0.1632	**0.1664**	7.01%		0.1418	0.1595	0.1590	0.1508	0.1795	**0.1818**	13.98%
		F1@5	0.3222	0.3052	0.2888	0.3452	0.3761	**0.4003**	15.96%		0.3722	0.3589	0.3724	0.3327	0.4283	**0.4303**	15.55%
		F1@10	0.2182	0.2320	0.2202	0.2568	0.2691	**0.2746**	6.93%		0.2322	0.2619	0.2602	0.2470	0.2940	**0.2978**	13.71%
		AUC	0.7817	0.8887	0.8614	0.9112	0.9164	**0.9178**	0.72%		0.7641	0.8923	0.8855	0.8768	0.9291	**0.9337**	4.64%
	T=2014	Rec@5	0.5142	0.5116	0.4665	0.5625	0.6121	**0.6577**	16.92%		0.6203	0.5768	0.5842	0.5494	0.6742	**0.6840**	10.27%
		Rec@10	0.5792	0.6661	0.6185	0.7198	0.7471	**0.7698**	6.95%		0.6570	0.7114	0.6927	0.6835	0.7952	**0.7998**	12.43%
		Prec@5	0.2508	0.2457	0.2284	0.2706	0.2962	**0.3148**	16.33%		0.2825	0.2586	0.2689	0.2529	0.3068	**0.3109**	10.05%
		Prec@10	0.1447	0.1636	0.1538	0.1776	0.1851	**0.1898**	6.87%		0.1510	0.1623	0.1611	0.1588	0.1840	**0.1850**	13.99%
		F1@5	0.3371	0.3320	0.3066	0.3654	0.3992	**0.4258**	16.53%		0.3882	0.3571	0.3683	0.3464	0.4217	**0.4275**	10.12%
		F1@10	0.2316	0.2627	0.2463	0.2849	0.2967	**0.3045**	6.88%		0.2455	0.2643	0.2614	0.2577	0.2989	**0.3005**	13.70%
		AUC	0.7359	0.8904	0.8619	0.9087	0.9112	**0.9206**	1.31%		0.7829	0.8834	0.8747	0.8770	0.9243	**0.9245**	4.65%

Methods Compared. As TaPEm is a task-guided heterogeneous network embedding framework, and we focus on the problem of author identification, we choose the following baselines.

(1) Feature engineering–based supervised method.
- **Sup**: Triggered by KDD Cup 2013, the problem of author identification has recently garnered attention, and top solutions of the challenge heavily relied on feature engineering followed by supervised ranking models on these features [8, 17]. Following them, we extract 16 features for each pair of paper and author in the training set. For more details about the features, refer to Table 2 of Zhang et al., [34]. As for the ranking model, we tried logistic regression, support vector machine, gradient boosting, random forest and multi-layer neural network (NeuN), and found that NeuN performed the best.

(2) General purpose heterogeneous network embedding method.
- **metapath2vec++ (MPV) [7]**: The state-of-the-art heterogeneous network embedding method based on meta-path guided random walk that learns general node embeddings. To directly compare within our setting, where papers in test data are not known during training, we represent a paper by the words of its abstract. We adopt a GRU–based PaperEncoder$(\cdot)$ shown in Eqn. 6 to encode a paper into a vector.

(3) Task-guided heterogeneous network embedding methods.
- **HNE [4]**: A task-guided heterogeneous network embedding method that resort to the network structure of an academic network rather than exploring the paper content.
- **Camel [34]**: The state-of-the-art heterogeneous network embedding–based method for author identification in which task-dependent and content-aware skip-gram model is proposed to formulate the correlations between each paper and its indirect author neighbors.
- **$\text{TaPEm}_{\text{npv}}$**: A variant of TaPEm in which instead of the pair validity classifier, a dot product is used. i.e., $\mathcal{L}_{\text{pv}}(v,u) = y_{v,u}\sigma(\mathbf{p}_v^T\mathbf{q}_u) + (1-y_{v,u})(1-\sigma(\mathbf{p}_v^T\mathbf{q}_u))$.

We do not compare with homogeneous network embedding methods, such as, DeepWalk [23] and node2vec [12], as their performance have been surpassed by methods designed for heterogeneous network embedding [7].

Evaluation Metrics. Recall that our task is to rank candidate authors for each paper $v \in I_{\geq T}$ in test dataset, where $I_{\geq T}$ denotes papers published after timestamp T. Hence, we use four commonly used ranking metrics, i.e., Recall@N, Precision@N, F1 score and AUC, to evaluate the performance of each method. Recall@N measures the ratio of the number of true authors among top-N predicted ranked list over the total number of true authors, Precision@N measures the ratio of the number of true authors in top-N predicted ranked list, and F1 measures the harmonic mean of precision and recall: F1 is high only if both precision and recall are high. AUC measures the probability of a positive instance being ranked higher than a randomly chosen negative one.

Experimental Settings. We use papers published before timestamp T 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 performance on validation data gives the best result, which is different from [34] that only has test datasets. For reliability, the reported results are averaged over 5 runs. Following [34], we simulated 5 walks from every node, and each walk is of length 20. We set the node embedding dimension $K = 128$, pair embedding dimension $d = 100$, margin $\xi = 0.1$, window size $\tau = 3$, dropout ratio to 0.15, and number of negative contexts $k = 1$. To show that TaPEm is a general framework that is not dependent on the selection of meta-paths, we only use a single meta-path "APA" for TaPEm, and compare with our baseline heterogeneous network embedding methods, i.e., metapath2vec++, HNE and Camel, that leverage multiple meta-paths, i.e., "APA", "APPA", and "APVPA". This is meaningful because useful meta-paths are usually task-dependent, and by relying on only a single meta-path, we demonstrate that our framework can be easily applied to various tasks. Following the setting of HNE and Camel, 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. For completeness, we also show evaluations on the whole authors set. For training efficiency, we pretrained the embeddings of TaPEm with those of Camel, but the accuracy is similar without the pretraining. The source code of TaPEm is available on github[3].

6.2 Performance Analysis

RQ 1) Author identification performance: Table 1 shows the author identification result of all the compared methods on both datasets in terms of various ranking metrics. We have the following observations. 1) TaPEm outperforms the baseline methods, especially when N is small, where N is the number of authors in the predicted list. This verifies that our pair embedding together with

[3] https://github.com/pcy1302/TapEM

Table 2: Author identification performance on relatively inactive users (#papers ≤ 5) (vs. Camel).

	T	Methods	Recall@N N =1	N =2	N =5	N =10	Precision@N N =1	N =2	N =5	N =10	F1@N N =1	N =2	N =5	N =10	AUC
AMiner-Top	2013	Camel	0.1808	0.3035	0.5012	0.6646	0.3155	0.2734	0.1887	0.1244	0.2299	0.2877	0.2742	0.2096	0.8854
		TaPEm	**0.2677**	**0.4131**	**0.6037**	**0.7220**	**0.4496**	**0.3697**	**0.2251**	**0.1360**	**0.3356**	**0.3902**	**0.3279**	**0.2289**	**0.8935**
		Improve.	48.06%	36.11%	20.45%	8.64%	42.50%	35.22%	19.29%	9.32%	45.98%	35.63%	19.58%	9.21%	0.91%
	2014	Camel	0.1624	0.2739	0.4831	0.6619	0.3372	0.2865	0.2094	0.1440	0.2192	0.2801	0.2922	0.2365	0.8909
		TaPEm	**0.2312**	**0.3670**	**0.5679**	**0.6900**	**0.4515**	**0.3759**	**0.2433**	**0.1507**	**0.3058**	**0.3714**	**0.3406**	**0.2473**	**0.8934**
		Improve.	42.36%	33.99%	17.55%	4.25%	33.90%	31.20%	16.19%	4.65%	39.51%	32.60%	16.56%	4.57%	0.28%

Table 3: Result comparisons on the whole authors set of AMiner-Top (T=2013) (over all authors and inactive authors).

Whole authors set						
Recall @N	All authors			Inactive authors (#papers ≤ 5)		
	Camel	TaPEm	Impr.	Camel	TaPEm	Impr.
N=10	0.0502	**0.1018**	102.79%	0.0304	**0.0749**	146.38%
N=30	0.1011	**0.1955**	93.37%	0.0631	**0.1438**	127.89%
N=50	0.1483	**0.2485**	67.57%	0.0943	**0.1871**	98.41%
N=100	0.2238	**0.3520**	57.28%	0.1461	**0.2755**	88.57%
N=200	0.3176	**0.4587**	44.43%	0.2213	**0.3651**	64.98%

the pair validity classifier can capture the fine-grained pairwise relationship between two nodes, which pushes true authors to the top ranks. 2) $\text{TaPEm}_{\text{npv}}$, which is a variant of TaPEm without the pair validity classifier, still outperforms other baselines. This verifies the benefit of our pair embedding framework itself over the skip-gram based node embedding methods. 3) From the comparisons between TaPEm and $\text{TaPEm}_{\text{npv}}$, we can verify that the pair validity classifier further improves the performance by encoding pair validity information into the pair embedding. Moreover, although not shown in the paper, it is important to note that TaPEm converges about 10 times faster than $\text{TaPEm}_{\text{npv}}$ in average, which shows another benefit of explicitly incorporating the pair validity classifier. 4) The performance of TaPEm is rather similar to that of Camel in terms of AUC, which is a metric that treats a mistake in the higher part of the ranked list as equal to one the lower part. Outperforming considerably in terms of position-aware metrics (e.g., Recall@N) while performing similar in terms of AUC implies that TaPEm focuses on the accuracy of the top-ranked authors at the expense of the accuracy in the lower part of the list, which is a desideratum for a ranking algorithm. 5) Embedding based methods generally perform better than the supervised learning–based method, indicating that the feature engineering process is error-prone. 6) Although metapath2vec++ is a general purpose embedding method, it generally outperforms HNE, which is specifically designed for author identification task. We attribute such result to the fact that metapath2vec++ is modified to integrate the paper content, while HNE only considers the keyword of a paper. This implies that paper content plays a critical role in identifying authors. 7) Table 3 shows the performance on the whole author candidate set of AMiner-Top dataset. We observe that the improvement of TaPEm is more significant than the performance on the sampled author candidate set shown in Table 1, which reaffirms the effectiveness of TaPEm.

RQ2) Performance on less active authors: Recall that the skip-gram based model is not appropriate for our task because it is biased to active authors (refer to our discussions on Figure 2), and thus performs poorly on inactive authors. However, we observe that most authors publish only few papers: in AMiner-Top and AMiner-All datasets, authors who published fewer than six papers constitute approximately 92%, which indicates that most authors are inactive. Therefore, identifying inactive authors in author identification task is in fact crucial, but challenging owing to the limited number of historical data. Table 2 shows the performance comparisons on relatively inactive authors, where we consider an author inactive if he/she has fewer than 6 publications. We observe that 1) TaPEm outperforms Camel, and the improvement is more significant than when considering all authors: for T=2013, the improvement is 15.33% on all authors but 20.45% on inactive authors in terms of Recall@5. 2) In Table 3 we also show that TaPEm considerably outperforms Camel when the candidate authors are the set of whole authors. From the above results on less active authors, we ascertain the effectiveness of the pair validity classifier in discriminating less active true authors.

RQ2-1) Qualitative analysis: To further demonstrate the effectiveness of TaPEm in identifying inactive authors, we conduct qualitative experiment on three papers published in 2006 (AMiner-Top 2013). Our goal here is to show that Camel, which is based on the skip-gram model, is indeed trained to be biased to active authors, whether or not they are the true author, because an active author is more likely to appear frequently together with a target paper within the same context window of random walks.

In Table 4a, we present a case where the most active author in AMiner dataset ("Jiawei Han" in bold) is in fact one of the true authors. In this case, Camel successfully ranked "Jiawei Han" in the first place as expected, whereas the other two relatively inactive authors are ranked far below the list. On the other hand, TaPEm showed better ranking performance regardless of the author's activeness. Table 4b shows a case where "Jiawei Han", who is not a true author, appeared most frequently with the query paper. In this case, Camel ranked "Jiawei Han" higher than the true authors even though he is not a true author. This behavior is expected as Camel is based on the skip-gram model that is biased to active authors. On the other hand, TaPEm again demonstrates its robustness to the activeness of authors. The last case in Table 4c shows a case where the most active author is a true author, and the second most active author in AMiner dataset, i.e., "Philip S. Yu", is a frequently appearing false author. In this case, Camel ranked "Philip S. Yu" higher than "Qiaozhu Mei" and "Dong Xin" due to his activeness. Moreover, we noticed that the rankings of Camel simply follow the order of the number of publications of the authors regardless of their authorships, which is a consequence of the skip-gram model. On the other hand, TaPEm is robust to the activeness of authors.

Table 4: Ranking of true authors and frequently appearing false authors for a query paper. Frequently appearing false authors denote authors that frequently appear together with the query paper within the same context window of random walks, but who are not the true authors.

(a) Case 1: True authors contain an active author.

Paper: (CIKM'06) Mining compressed commodity workflows from massive RFID datasets			
	Author (num. publications)	Rank	
		Camel	TaPEm
True authors	**Jiawei Han** (141)	1	8
	Xiaolei Li (12)	198	1
	Hector Gonzalez (9)	296	81
Frequently appearing false authors	Yizhou Sun (23)	94	418
	Jae-Gil Lee (10)	323	196
	John Paul Sondag (1)	1043	3650

(b) Case 2: Frequently appearing authors contain an active author.

Paper: (KDD'06) A mixture model for contextual text mining			
	Author (num. publications)	Rank	
		Camel	TaPEm
True authors	Cheng Xiang Zhai (51)	4	3
	Qiaozhu Mei (21)	24	4
Frequently appearing false authors	**Jiawei Han** (141)	2	122
	Yintao Yu (6)	601	372

(c) Case 3: Both author groups contain an active author.

Paper: (KDD'06) Generating semantic annotations for frequent patterns with context analysis			
	Author (num. publications)	Rank	
		Camel	TaPEm
True authors	**Jiawei Han** (141)	1	14
	Qiaozhu Mei (21)	44	9
	Dong Xin (20)	130	26
Frequently appearing false authors	**Philip S.Yu** (122)	7	41
	Xifeng Yan (36)	15	19
	Charu C.Aggarwal (30)	16	303

Table 5: Average rank violation comparisons.

	Camel	TaPEm
Average rank violation (the smaller the better)	3.8374	0.4519

To quantitatively show the reliability of the above results, we compare the ranking of true authors and frequently appearing false authors of each paper, and see whether true authors are indeed ranked higher than frequently appearing false authors. More precisely, every time a frequently appearing false author is ranked higher than a true author, we count it as a violation. For each paper, we sampled the top-N most frequently appearing false authors for comparisons, where N is the number of the actual authors. Table 5 shows the comparisons of average rank violation count. We observe that the average rank violation count of TaPEm is about 8 times less than that of Camel, which again demonstrates the effectiveness of our pair embedding framework.

RQ 3) Ablation study: To measure the impact of each component of TaPEm on the author identification accuracy, we conduct ablation studies in Table 6. We have the following observations: 1) Each component of TaPEm, i.e., Dropout, and attentive pooling, contributes to the performance of TaPEm. 2) A simple linear combination of a context path instead of modeling it with a BiGRU as in Eqn. 8 performs worse. This implies that it is helpful to model a context path as a sequence. 3) Nevertheless, even without these components, TaPEm still considerably outperforms the strongest baseline on AMiner-Top, which is Camel, implying the superiority of our novel pair embedding framework. 4) $\text{TaPEm}_{\text{pv+dot}}$ is equivalent to TaPEm, but the only difference is that the final prediction is done by a dot product between paper and author embeddings as done in Camel, instead of by the output of the pair validity classifier. We observe that $\text{TaPEm}_{\text{pv+dot}}$ still outperforms other baselines in Table 1, which implies that the pair validity classifier is also helpful for generating more accurate paper and author embeddings. 5) We observe that TaPEm outperforms $\text{TaPEm}_{\text{pv+dot}}$ more significantly for top-ranked authors. i.e., for small N of recall@N. This implies that the pair validity classifier helps distinguish valid pairs from invalid ones, which result in pushing true authors to the top ranks. In other words, the performance for top-ranked authors suffer without the pair validity classifier. 6) The performance of $\text{TaPEm}_{\text{npv}}$ that performs the worst among the ablations of TaPEm reaffirms the benefit of the pair validity classifier.

Table 6: Result for ablations of TaPEm.

	Ablations	Rec@1	Rec@2	Rec@5	Rec@10
AMiner-Top (T=2013)	No Dropout	0.2979	0.4515	0.6567	0.7766
	No attention	0.3024	0.4627	0.6679	0.7800
	No Eqn. 8	0.2824	0.4499	0.6623	0.7792
	$\text{TaPEm}_{\text{npv}}$	0.2755	0.4266	0.6405	0.7677
	$\text{TaPEm}_{\text{pv+dot}}$	0.2858	0.4392	0.6614	**0.7906**
	TaPEm	**0.3118**	**0.4823**	**0.6807**	0.7849

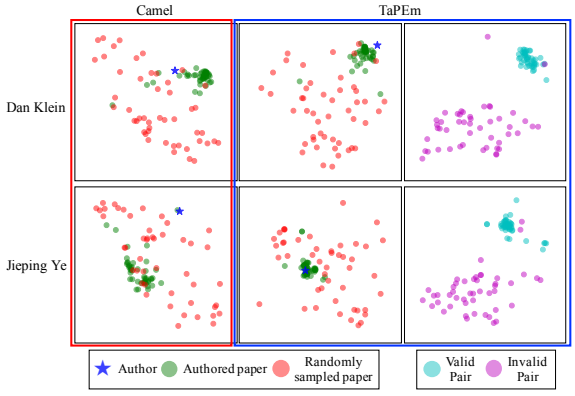


Figure 6: t-SNE visualization of author, paper and pair embeddings for two authors: Dan Klein and Jieping Ye.

RQ 4) Visualization of embeddings: To provide a more intuitive understanding of pair embeddings, we visualize paper, author embeddings and paper–author pair embeddings of two authors from AMiner-Top dataset by using t-SNE [19]. More precisely, for Camel, we plot an author along with the papers written by the author, and we also plot randomly sampled papers that are not written by the author as many as the number of authored papers. We also plot both paper/author embeddings, and pair embeddings of TaPEm. Note that the same set of authored papers and randomly sampled papers used for Camel are used for constructing the pair embeddings.

Compared with Camel, we observe in Figure 6 that the embeddings of authored papers of TaPEm are more tightly grouped together than those of Camel, and the author embedding of TaPEm is

placed relatively closer to the cluster of the authored papers than those of Camel. This implies that TaPEm generates more accurate representations of paper and author than Camel by making two nodes that constitute a pair close to each other not only if they are related to a similar research topic, but also the pair itself is valid at the same time; this is corroborated by the result of $\text{TaPEm}_{\text{pv+dot}}$ in Table 6 that outperforms Camel in Table 1. Moreover, we observe from the rightmost figure that when an author is coupled with the papers (both authored papers and randomly sampled papers) to form paper–author pair embeddings, it becomes easier to distinguish whether a pair is valid or not, which is the benefit obtained from the pair validity classifier. This is a useful for task-guided heterogeneous network embedding whose ultimate goal is to model the likelihood of pairwise relationship between two nodes.

6.2.1 Discussion. In the experiments, we focused on the problem of author identification as an application of our proposed framework. Under the space limitation, our intention is to delve deep into showing the effectiveness of TaPEm in various aspects, instead of covering many tasks but with limited experiments. However, we postulate that the final objective function $\mathcal{L}$ in Eqn. 15 can be leveraged to solve various real-world tasks whose the ultimate goal is to model the pairwise relationship between node v and u; user-item relationship in recommendation, paper-paper relationship in citation recommendation, and author-author relationship in collaborator recommendation.

7 CONCLUSION

In this paper, we proposed a novel task-guided pair embedding framework in heterogeneous network embedding that is useful for tasks whose goal is to predict the likelihood of pairwise relationship between two nodes. Instead of learning general purpose node embeddings, we directly focus on the pairwise relationship between two nodes that we are interested in, and learn the pair embedding considering its associated context path between the pair of nodes. Our pair validity classifier is effective in identifying less active true authors, and pushing true authors to the top ranks, which is desideratum for a ranking algorithm. As future work, we plan to investigate on the applicability of TaPEm on different tasks.
Acknowledgment: 2016R1E1A1A01942642, and SW Starlab (IITP-2018-0-00584).

REFERENCES

[1] Smriti Bhagat, Graham Cormode, and S Muthukrishnan. 2011. Node classification in social networks. In *Social network data analytics*. Springer.

[2] Rebecca M Blank. 1991. The effects of double-blind versus single-blind reviewing: Experimental evidence from the American Economic Review. *The American Economic Review* (1991).

[3] Shiyu Chang, Wei Han, Jiliang Tang, Guo-Jun Qi, Charu C Aggarwal, and Thomas S Huang. 2015. Heterogeneous network embedding via deep architectures. In *KDD*. ACM.

[4] Ting Chen and Yizhou Sun. 2017. Task-guided and path-augmented heterogeneous network embedding for author identification. In *WSDM*. ACM, 295–304.

[5] Ting Chen, Lu-An Tang, Yizhou Sun, Zhengzhang Chen, and Kai Zhang. 2016. Entity embedding-based anomaly detection for heterogeneous categorical events. *arXiv preprint arXiv:1608.07502* (2016).

[6] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).

[7] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *KDD*. ACM.

[8] Dmitry Efimov, Lucas Silva, and Benjamin Solecki. 2013. Kdd cup 2013-author-paper identification challenge: second place team. In *Proceedings of the 2013 KDD Cup 2013 Workshop*. ACM.

[9] Tao-yang Fu, Wang-Chien Lee, and Zhen Lei. 2017. Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning. In *CIKM*. ACM.

[10] Claire Le Goues, Yuriy Brun, Sven Apel, Emery Berger, Sarfraz Khurshid, and Yannis Smaragdakis. 2017. Effectiveness of anonymization in double-blind review. *arXiv preprint arXiv:1709.01609* (2017).

[11] Palash Goyal, Homa Hosseinmardi, Emilio Ferrara, and Aram Galstyan. 2018. Capturing edge attributes via network embedding. *IEEE Transactions on Computational Social Systems* 5, 4 (2018).

[12] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *KDD*. ACM.

[13] Xiao Huang, Jundong Li, and Xia Hu. 2017. Accelerated attributed network embedding. In *SDM*. SIAM.

[14] Xiao Huang, Jundong Li, and Xia Hu. 2017. Label informed attributed network embedding. In *WSDM*. ACM.

[15] Rana Hussein, Dingqi Yang, and Philippe Cudré-Mauroux. 2018. Are Meta-Paths Necessary?: Revisiting Heterogeneous Graph Embeddings. In *CIKM*. ACM.

[16] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[17] Chun-Liang Li, Yu-Chuan Su, Ting-Wei Lin, Cheng-Hao Tsai, Wei-Cheng Chang, Kuan-Hao Huang, Tzu-Ming Kuo, Shan-Wei Lin, Young-San Lin, Yu-Chen Lu, et al. 2015. Combination of feature engineering and ranking models for paper-author identification in KDD Cup 2013. *JMLR* 16, 1 (2015).

[18] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025* (2015).

[19] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *JMLR* (2008).

[20] Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in Pre-Training Distributed Word Representations. In *LREC*.

[21] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NeurIPS*.

[22] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. 2016. Asymmetric transitivity preserving graph embedding. In *KDD*. ACM.

[23] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *KDD*. ACM.

[24] Jason Priem. 2013. Scholarship: Beyond the paper. *Nature* 495, 7442 (2013), 437.

[25] Chuan Shi, Binbin Hu, Wayne Xin Zhao, and S Yu Philip. 2019. Heterogeneous information network embedding for recommendation. *IEEE Transactions on Knowledge and Data Engineering* 31, 2 (2019).

[26] Yu Shi, Huan Gui, Qi Zhu, Lance Kaplan, and Jiawei Han. 2018. Aspem: Embedding learning by aspects in heterogeneous information networks. In *SDM*. SIAM.

[27] Yu Shi, Qi Zhu, Fang Guo, Chao Zhang, and Jiawei Han. 2018. Easing embedding learning by comprehensive transcription of heterogeneous information networks. In *KDD*. ACM.

[28] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *JMLR* 15, 1 (2014).

[29] Yizhou Sun, Rick Barber, Manish Gupta, Charu C Aggarwal, and Jiawei Han. 2011. Co-author relationship prediction in heterogeneous bibliographic networks. In *ASONAM*. IEEE.

[30] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S Yu, and Tianyi Wu. 2011. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *Proceedings of the VLDB Endowment* 4, 11 (2011).

[31] Yizhou Sun, Yintao Yu, and Jiawei Han. 2009. Ranking-based clustering of heterogeneous information networks with star network schema. In *KDD*. ACM.

[32] Hongwei Wang, Fuzheng Zhang, Min Hou, Xing Xie, Minyi Guo, and Qi Liu. 2018. Shine: Signed heterogeneous information network embedding for sentiment link prediction. In *WSDM*. ACM.

[33] Xiao Wang, Peng Cui, Jing Wang, Jian Pei, Wenwu Zhu, and Shiqiang Yang. 2017. Community preserving network embedding. In *AAAI*.

[34] 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*.

[35] Daokun Zhang, Jie Yin, Xingquan Zhu, and Chengqi Zhang. 2018. Meta-Graph2Vec: Complex Semantic Path Augmented Heterogeneous Network Embedding. In *PAKDD*. Springer.

[36] Muhan Zhang and Yixin Chen. 2018. Link prediction based on graph neural networks. In *NeurIPS*.

[37] 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*, Vol. 2.