The following publication L. Xu et al., "MixSp: A Framework for Embedding Heterogeneous Information Networks With Arbitrary Number of Node and Edge Types," in IEEE Transactions on Knowledge and Data Engineering, vol. 33, no. 6, pp. 2627-2639, 1 June 2021 is available at https://doi.org/10.1109/TKDE.2019.2955945

MixSp: A Framework for Embedding Heterogeneous Information Networks with Arbitrary Number of Node and Edge Types

Linchuan Xu, Jing Wang, Lifang He, Jiannong Cao, *Fellow, IEEE,* Xiaokai Wei, Philip S. Yu, *Fellow, IEEE*
Kenji Yamanishi, *Member, IEEE*

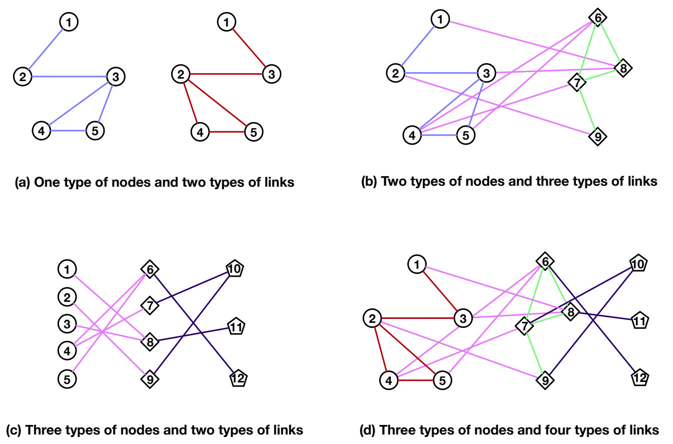
Abstract—Heterogeneous information network (HIN) embedding is to encode network structure into node representations with the heterogeneous semantics of different node and edge types considered. However, since each HIN may have a unique nature, e.g., a unique set of node and edge types, a model designed for one type of networks may not be applicable to or effective on another type. In this paper, we thus attempt to propose a framework for HINs with arbitrary number of node and edge types. The proposed framework constructs a novel mixture-split representation of an HIN, and hence is named as MixSp. The mixture sub-representation and the split sub-representation serve as two different views of the network. Compared with existing models which only learn from the original view, MixSp thus may exploit more comprehensive information. Node representations in each view are learned by embedding the respective network structure. Moreover, the node representations are further refined through cross-view co-regularization. The framework is instantiated in three models which differ from each other in the co-regularization. Extensive experiments on three real-world datasets show MixSp outperforms several recent models in both node classification and link prediction tasks even though MixSp is not designed for a particular type of HINs.

Index Terms—Network embedding, heterogeneous information networks, multi-label classification, link prediction.

1 Introduction

Network embedding [1] [2] [3] is to learn low-dimensional and dense node representations that encode the usually high-dimensional and sparse network representation, which has been extensively studied recently due to the effectiveness of the representations in many data mining tasks including node classification and link prediction. Network embedding is basically achieved by enforcing nodes close in the network representation to be close in a particular Euclidean space of interest. This paper studies heterogenous information networks (HINs) which usually consist of more than one type of nodes and edges [4] [5]. Besides the network structure, heterogeneous information network embedding needs to consider heterogeneous semantics of different node and edge types.

Each existing embedding model is usually network-specific. MTNE [6] is designed for networks with a single type of nodes but with multiple types of edges like Fig. 1(a).



(a) One type of nodes and two types of links
(b) Two types of nodes and three types of links
(c) Three types of nodes and two types of links
(d) Three types of nodes and four types of links

Fig. 1: Heterogeneous information networks with different number of node and edge types.

This type of networks is also studied in DMNE [7]. Besides, DMNE is applicable to Fig. 1(b), which is a coupled heterogeneous network studied in EOE [8]. PTE [9] is designed for heterogeneous bipartite networks where all sub-bipartite networks share a set of nodes, e.g., a heterogeneous bipartite network consisting of two sub-bipartite networks in Fig. 1(c). metapath2vec [10] and HEER [11] are mainly designed for other networks which can have multiple types of nodes and edges like Fig. 1(d). Although these models have been demonstrated very effective on the respective networks, they may not be applicable to or effective on other types of networks due to their network-specific modeling. Note that there may be many other types of networks since the

- *Linchuan Xu, Jing Wang and Kenji Yamanishi are with the Department of Mathematical Informatics, Graduate School of Information Science and Technology, The University of Tokyo, Hongo, 7-3-1 Bunkyoku, Tokyo, 113-8656, JAPAN.*
  E-mail: {linchuan_xu, jing_wang, yamanishi}@mist.i.u-tokyo.ac.jp
- *Lifang He is with Department of Computer Science & Engineering, Lehigh University, Bethlehem, PA 18015, USA.*
  E-mail: lifanghescut@gmail.com
- *Jiannong Cao is with Department of Computing, the Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong.*
  E-mail: csjcao@comp.polyu.edu.hk
- *Xiaokai Wei is with Facebook Inc., 1 Hacker Way, Menlo Park, CA, USA.*
  E-mail: weixiaokai@gmail.com
- *Philip S. Yu is with Department of Computer Science, University of Illinois at Chicago, IL, 60601, USA.*
  E-mail: psyu@uic.edu
- *Jing Wang is the corresponding author.*

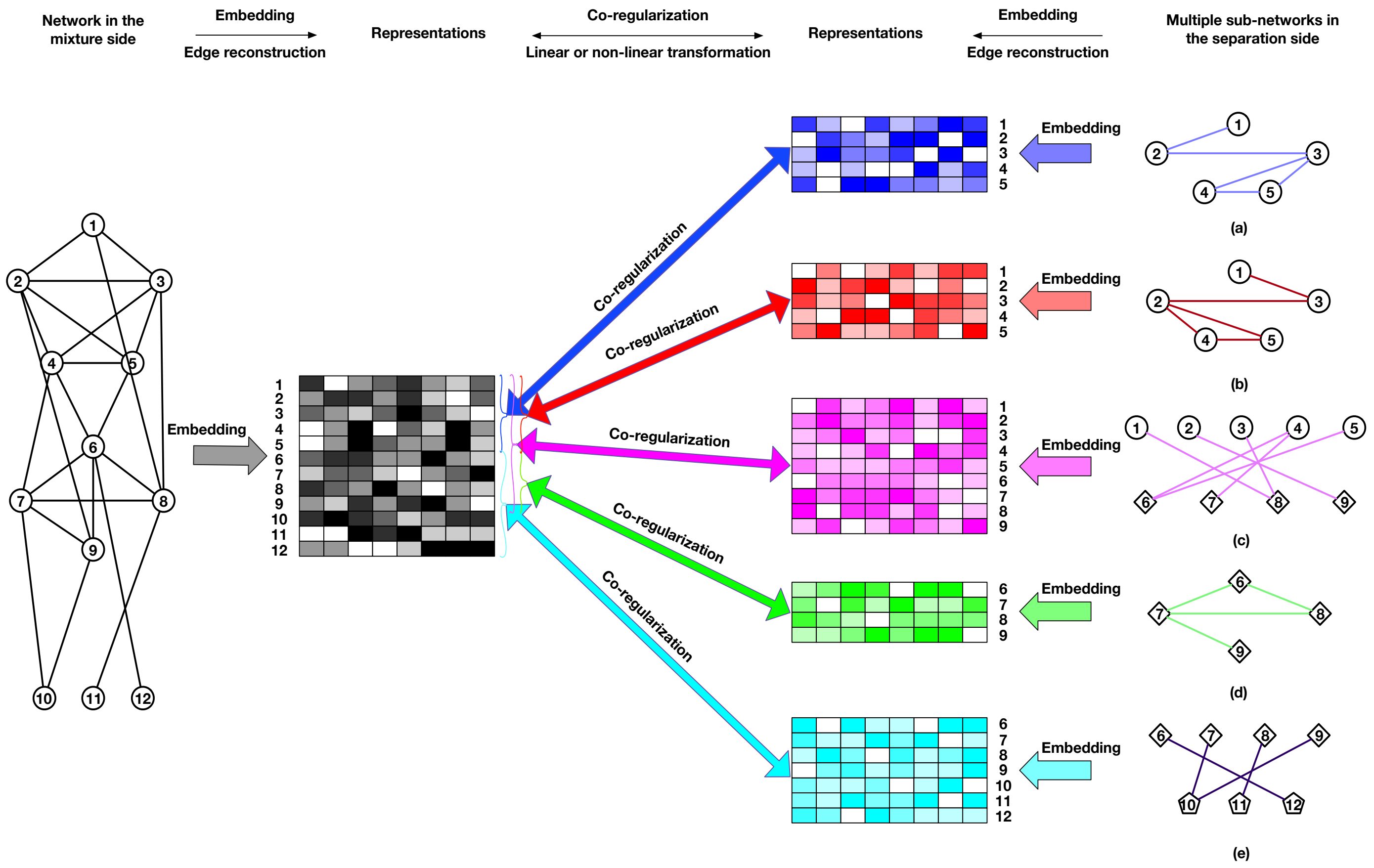


Fig. 2: The framework of embedding the mixture-split representation of an HIN where each row of the matrices represents a node representation. Note that the co-regularization is only enforced on the corresponding nodes between the mixture view and the split view as indicated by the colorful brackets, which is explained in Section 3.4.

number of node and edge types and the semantic meanings of nodes and edges can vary in practice. Hence, it may need great efforts on studying the characteristics of each type of heterogeneous information networks and designing a tailored embedding model.

Before introducing the proposed framework, we briefly explain the logical thinking behind it. Recap that the objective of heterogeneous network embedding is to explicitly model the heterogeneities of different node and edge types while embedding the network structure, but the network-specific modeling would limit the applicability. Hence, it is a natural thought that a general solution should preserve the heterogeneities and avoid the network-specific modeling simultaneously. One potential way is to disentangle the two objectives into two models. To avoid the network-specific modeling, heterogeneous nodes and edges should be treated as homogeneous such that any homogeneous network embedding method is applicable. To preserve the heterogeneities, a sub-network can be constructed with respect to each type of edges and the corresponding nodes, and then all sub-networks are embedded independently. In this way, two copies of the given HIN are constructed and two sets of node representations can be obtained. The direct combination of the two sets of node representations is a naive way to have both objectives achieved. But here, we propose to jointly learn the two sets of node representations and the motivation is presented below.

In particular, we design a new representation for any HIN, which is named as a mixture-split representation. In the mixture sub-representation, all the nodes and edges are present, and multiple types of edges (if exist) among a single pair of nodes are merged into a weighted edge. In the split sub-representation, the HIN is split into multiple sub-networks where each sub-network only consists of one type of edges and the corresponding nodes.

As mentioned above, the network in the mixture sub-representation should be embedded as a homogeneous network in order to avoid the network-specific modeling. In this way, however, the node representations are less optimal since heterogeneous semantics are ignored. Besides, the merging of multiple edges into a weighted one causes information loss. To alleviate the two drawbacks, node representations learned from the split sub-representation can be utilized since they together preserve all the heterogeneous information, which is the motivation for the joint learning of the two sets of node representations.

The proposed idea is illustrated in Fig. 2. Each node has one representation corresponding to the mixture network while the number of representations in the split network is the number of types of edges it is involved in (i.e., the number of sub-networks it belongs to). The node representations in the mixture network, on the one hand, are regularized by its network structure via network embedding. On the other hand, they are regularized by the corresponding node representations in the split network in order to capture the heterogeneous characteristics brought by different types of edges. Note that node representations in the split network can also be improved by the node representations in the mixture network. Each node representation in the split network only encodes the corresponding sub-network structure, but it is better for each representation to have access to other related sub-network information, which is essentially the motivation for jointly considering multiple types of nodes and edges together as a heterogeneous information network.

More concretely, for data mining tasks regarding a particular type of edges, e.g., link prediction, it is better to employ the corresponding node representations in the split network instead of those in the mixture network. It is likely that the other types of edges can be used as references [1]. But node representations in the mixture network explicitly encode all types of edges, and hence other types of edges may play a role as important as the type of interest in the link prediction, which is not appropriate since each type of edges has a unique nature. Instead, node representations in the split network only explicitly encode the targeted type of edges and implicitly take the other types of edges into consideration via the regularization from the node representations in the mixture network. Here, it is worth mentioning that node representations in the mixture network are appropriate for applications that are not regarding a particular type of edges, e.g., node classification.

Besides the advantages in terms of generalization, the proposed solution may also outperform existing models on the data mining tasks. For node classification, since node representations in the mixture network encode all types of edges jointly and node representations in the split network encode all types of edges separately, the edge information are exploited more comprehensively. From another point of view, the mixture network represents one view of the heterogeneous information network, and the split network which consists of multiple sub-networks represents another view. The two views have different network structures, and thus jointly embedding the two views may obtain more comprehensive information than existing models that only exploit the original view, which has actually been the motivation for constructing multiple views from a single image for image processing tasks [12]. As for link prediction, some existing models, e.g., metapath2vec [10], learn node representations that explicitly encode all type of edges, which is not appropriate as mentioned above. Our experiments have also verified this point.

We design a mixture-split (MixSp) framework based on the proposed idea, and demonstrate the framework in three models which differ from each other in the co-regularization. The main contributions of the paper are summarized as follows:

- To our best knowledge, this paper is the first attempt to propose a solution for embedding HINs with arbitrary number of node and edge types such that efforts on the analysis of network characteristics and the network-specific modeling can be largely saved.
- The paper proposes a novel mixture-split representation of any type of HINs. The mixture sub-representation and the split sub-representation serve as two different views of an HIN, and then more comprehensive information may be obtained than from the original view.
- The paper proposes three models which utilize different mechanisms as the cross-view co-regularization to refine node representations in both views.
- Through experiments on three real-world datasets, MixSp is shown to be more effective than recent models in both link prediction and multi-label node classification tasks.

1. Different types of edges may have conflicting semantics in some cases. But usually only non-conflicting edges are jointly studied in existing work and so does this paper.

The rest of the paper is organized as follows. Section 2 presents the related work. Section 3 develops the proposed models. In Section 4, empirical evaluation is presented. We conclude and introduce our future directions in Section 5.

2 Related Work

2.1 Homogeneous Network Embedding

One popular stream of studies, such as Deepwalk [1], node2vec [3], struc2vec [13], are based on Skip-gram [14] which is a language model designed for word embedding. The adoption of the language model is based on the connection established between the sequence of nodes obtained by random works on the network representation and the sentence of words. These Skip-gram based models have been proved to be equal to matrix factorization, and many variants of the matrix factorization have been designed, such as TADW [15], GraRep [16], and NetMF [17].

There are many other kinds of studies. LINE [2] estimates the probability of each edge, and then learns the representations by minimizing the difference between the estimated probabilities and the empirical probabilities. M-NMF [18] preserves both the edge relationships and community relationships among nodes via matrix factorization. MSRE [19] assumes that each participant plays multiple roles in a network, and then learns multiple role-specific embeddings for each node. Cetera [20] constructs cliques with different number of nodes, and jointly embeds all the cliques by kernelized multi-dimensional array factorization. Besides preserving the similarity among nodes, RaRE [21] further explores social rank where a high rank of a node is due to its popularity. Deep learning models [22] [23] have been developed to model high-order non-linear relationships. The recent popular min-max framework proposed in generative adversarial nets (GAN) [24] has been adopted in network embedding [25] since it can jointly learn a discriminative model and a generative model. LATTE [26] proposes application-oriented embedding which incorporates subsequent applications into network embedding. Besides learning node representations, DeepGL [27] also learns relational functions that generalize for computation on any arbitrary graph. ICANE [28] embeds not only edges but also edge content which contains rich information about the interaction environment. EII [29] employs node content to explore two different reasons for nodes to interact with each other. With the observation that some nodes only have edges, and some nodes only have node content, CVLP [30] proposes a consensus embedding mechanism to fuse edge information and content information into a common space for cross-view link prediction.

2.2 Heterogeneous Network Embedding

Since there are various types of HINs, we review the heterogeneous network embedding according to the type of networks studied. Here, we roughly classify HINs into three categories, which are HINs with emphasis on one type of nodes involved in multiple types of edges, e.g., Fig. 1(a), ones with emphasis on two types of nodes involved in one type of edges, e.g., Fig. 1(b) and Fig. 1(c), and ones with generally multiple types of nodes and multiple types of edges, e.g., Fig. 1(d). The three types of networks are referred to as Type I, Type II, and Type III, respectively, for brevity. Note that the point of proposing the three types is to emphasize that there are no existing models applicable to all types of HINs even though the definitions of the three types may not be precise.

2.2.1 Methods for Type I

A single pair of nodes can have multiple different types of relationships, e.g., similarities, social relations, interactions, information flows as summarized in [31]. A homogeneous sub-network can be constructed for each type of edges, but it is usually better to jointly consider multiple types since it is likely that two nodes with one type of relationships may also have another type of relations. Hence, the major issue for embedding this type of networks is how to utilize information of one type of edges to improve the learning from another type. To this end, MTNE [6] proposes two different mechanisms. One mechanism is to enforce a common representation which is shared by all the sub-networks like multi-task metric learning [32] while the other is to enforce a consensus representations on which all subnetwork-specific representations should agree. DMNE [7] proposes another two different mechanisms. In particular, DMNE learns subnetwork-specific representations for each sub-network and performs co-regularization on each pair of subnetwork-specific representations. Multi-view network embedding methods, e.g., MVE [33], mve2vec [34], and MVNE [35], regard each type of edges as a view, and then jointly embed multiple views via co-regularization on the node representations as well.

2.2.2 Methods for Type II

Heterogeneous networks with emphasis on two types of nodes involved in one type of edges are mainly those networks with bipartite representations. In this paper, we consider heterogeneous bipartite networks with more than one type of edges. There are different scenarios in practice. PTE [9] studies three bipartite networks where all the bipartite networks share one set of nodes, e.g., word-word, word-document, and word-label bipartite networks share the set of words as nodes. PTE embeds each bipartite network like LINE(2nd) [2] and connects the embedding of multiple bipartite networks through enforcing the same representation for each node in the word set. Besides the edges connecting two types of nodes, each type of nodes may also have edges among themselves, e.g., papers can have citation relationships, which is studied in EOE [8]. EOE learns node representations by embedding each sub-network consisting of one type of nodes as a homogeneous network, and then further embeds node representations to encode the bipartite edges. The network studied in EOE can be viewed as a two-layered network. Besides, there may exist multi-layered networks. To embed the multi-layered networks, MANE [36] embeds each sub-network consisting of one type of nodes, i.e., each layer, as a homogeneous network, and then models the inter-layer edges as interactions in collaborative filtering. Note that DMNE [7] is also applicable to this type of networks with multi-layers.

2.2.3 Methods for Type III

Other types of HINs do not have salient characteristics in terms of the number of node and edge types, and thus are categorized into this group. A steam of models for embedding this type of networks are based on random walks aware of heterogeneous edges and nodes for obtaining meta-paths which are usually pre-defined sequences of types of nodes and then employ neural network models like Skip-gram [14] for encoding the meta-paths into node representations, such as metapath2vec [10], Hin2Vec [37], ESim [38], and [39]. There are also several other models not based on meta-paths, e.g., HNE [40], DIME [41], MINES [42], ASPEM [43] and HEER [11]. Besides learning node representations, HEER [11] learns edge representations such that edges connecting different types of nodes are comparable when edge representations are used as transcriptions.

2.3 The Relevance of MixSp to Existing Work

Different from all aforementioned heterogeneous network embedding methods which are designed for a particular type of networks, MixSp is general-purpose. The advantage of the wide applicability results from the novel mixture-split representation. MixSp is also different from all multi-view embedding methods because MixSp constructs two views of a single network instead of regarding each type of edges as a view.

3 METHODOLOGY

3.1 Preliminaries

DEFINITION 1. (HIN) *A heterogeneous information network (HIN) is denoted as $G(N, E)$. N is a set of nodes with a node type mapping function $\phi : N \to T$. E is a set of weighted or unweighted, directed or undirected edges with an edge type mapping function $\psi : E \to R$. Note that $|T| >= 1$, $|R| >= 2$, each node may be involved in multiple types of edges, and each edge may connect either two nodes of the same type or two nodes of different types.*

The type of nodes is defined as the type of entities which the nodes represent, and the type of edges is defined for a particular pair of node types, e.g., author, paper and conference venue are different node types in a heterogeneous bibliographic information network with co-authorship and authorship as different edge types [4] [5]. The definition can be applicable to all HINs studied in network embedding but is not appropriate for a general solution as mentioned in the introduction. To facilitate a general solution, we thus further define a mixture-split representation of a heterogeneous information network $G(N, E)$. In particular, the network in the mixture view and sub-networks in the split view are defined as follows:

DEFINITION 2. (MIXTURE VIEW) *The network in the* ***mixture*** *view is denoted as $G^m(N^m, E^m)$. N^m is the same set of nodes as N but with all nodes treated as the same type. E^m is a homogeneous set of weighted or unweighted, directed or undirected edges modified from E. The modification is that all different types of edges are treated as the same type, and multiple edges (if exist) between each pair of nodes in $G(N, E)$ are merged into a weighted edge.*

DEFINITION 3. (SPLIT VIEW) *A sub-network for each edge type $r \in R$ in the* ***split*** *view is denoted as $G^s_r(N^s_r, E^s_r)$ if it only consists of one type of nodes or $G^s_r(N^s_{r_A} \cup N^s_{r_B}, E^s_r)$ if it consists of two types of nodes. N^s_r or $N^s_{r_A} \cup N^s_{r_B}$ is the sub-set of nodes involved in the edge type r where $\phi(N^s_{r_A}) \neq \phi(N^s_{r_B})$. E^s_r is a sub-set of E with each $\psi(e^s_{(i,r)}) = r$.*

Note that m and s as the superscripts denote the mixture view and the split view, respectively. Now, an HIN $G(N, E)$ is represented by a network $G^m(N^m, E^m)$ in the mixture view and multiple sub-networks in the split view. $G^m(N^m, E^m)$ is treated as a homogeneous network and each sub-network is either a homogeneous network $G^s_r(N^s_r, E^s_r)$ or a bipartite network $G^s_r(N^s_{r_A} \cup N^s_{r_B}, E^s_r)$. In MixSp, the $G^m(N^m, E^m)$ and the multiple sub-networks are embedded instead of the original $G(N, E)$.

3.2 Embedding of the Mixture View

Since the focus of the paper is heterogeneous information networks, to embed the structure of $G^m(N^m, E^m)$, we directly employ existing homogeneous network embedding techniques. We employ the idea of LINE(1st) [2] which encodes the first-order linkage relationships among the nodes into node representations. In particular, LINE(1st) estimates the probability of an edge between each pair of nodes through node representations, and then learns node representations such that the Kullback−Leibler (KL) divergence between the estimated probability distribution of the edges and the empirical distribution is minimized.

The estimated probability of an edge is quantified as follows:

$$p(\boldsymbol{n}^m_i, \boldsymbol{n}^m_j) = \sigma((\boldsymbol{n}^m_i)^\top \boldsymbol{n}^m_j) = \frac{1}{1 + \exp\{-(\boldsymbol{n}^m_i)^\top \boldsymbol{n}^m_j\}}, \quad (1)$$

where $\boldsymbol{n}^m_i \in \mathbb{R}^{d_m}$ and $\boldsymbol{n}^m_j \in \mathbb{R}^{d_m}$ are column-vector representations of node i and node j in the mixture view, respectively, and $d_m \in \mathbb{R}$ is the dimension. $\sigma(\cdot)$ is the sigmoid function. The corresponding empirical probability can be computed as $\hat{p}(i, j) = \frac{w^m_{ij}}{W}$ where w^m_{ij} is the weight of the edge and $W = \sum_{(i,j)\in E^m} w^m_{ij}$.

The loss function of minimizing the KL divergence after straightforward calculations can be obtained as follows:

$$\mathcal{L}^m = -\sum_{(i,j)\in E^m} w^m_{ij} \log \sigma((\boldsymbol{n}^m_i)^\top \boldsymbol{n}^m_j), \quad (2)$$

To further avoid a trivial solution that $n^m_{ik} = \infty$ for $i = 1, ..., |N_m|$ and $k = 1, ..., d_m$, LINE(1st) samples multiple negative edges for each node. For more details about how to derive the loss function, one may refer to the original paper [2].

3.3 Embedding of the Split View

For each homogeneous sub-network $G^s_r(N^s_r, E^s_r)$, it is also embedded via the first-order linkage encoding as demonstrated above. For a bipartite sub-network $G^s_r(N^s_{r_A} \cup N^s_{r_B}, E^s_r)$, we employ the idea of LINE(2nd) which encodes the second-order linkage relationships among the nodes [2]. In particular, LINE(2nd) estimates the conditional probability of each node of one type generated by a particular node of another type through node representations, and then learns node representations such that the KL divergence between the estimated conditional probability distribution and the empirical distribution is minimized.

The conditional probability of node i in $N^s_{r_A}$ generated by node j in $N^s_{r_B}$ is defined as follows:

$$p(\boldsymbol{n}^s_{(r_A,i)}|\boldsymbol{n}^s_{(r_B,j)}) = \frac{\exp\{(\boldsymbol{n}^s_{(r_A,i)})^\top \boldsymbol{n}^s_{(r_B,j)}\}}{\sum_{i'\in r_A} \exp\{(\boldsymbol{n}^s_{(r_A,i')})^\top \boldsymbol{n}^s_{(r_B,j)}\}}, \quad (3)$$

where $\boldsymbol{n}^s_{(r_A,i)} \in \mathbb{R}^{d_s}$ and $\boldsymbol{n}^s_{(r_B,j)} \in \mathbb{R}^{d_s}$ are node representations of node i and node j, respectively. The corresponding empirical probability is calculated as $\hat{p}(i|j) = \frac{w^s_{(r,ij)}}{deg_j}$ where $w^s_{(r,ij)}$ is the weight of the edge and deg_j is the degree of node j defined as $deg_j = \sum_i w^s_{(r,ij)}$.

The loss function of minimizing the KL divergence can be obtained as follows:

$$\mathcal{L}^s_r = -\sum_{(i,j)\in E^s_r} w^s_{(r,ij)} \log p(\boldsymbol{n}^s_{(r_A,i)}|\boldsymbol{n}^s_{(r_B,j)}), \quad (4)$$

Since Eq. (3) has to be summarized over the entire set of nodes in $N^s_{r_A}$, it is may not be efficient when dealing with large-scale networks. Hence, LINE(2nd) also employs negative sampling [2].

3.4 Co-regularization

The co-regularization is enforced between node representations in the split view and the corresponding representations in the mixture view. As illustrated in Fig. 2, the node representations of the sub-network Fig. 2(a) have and only have the co-regularization relationship with representations of node 1 through node 5 in the mixture view. This is because the sharing of the same entities can be appropriate relationships for establishing the co-regularization but the representations of other nodes in the mixture view have no direct relationships with node representations of the sub-network. Besides, the representations of node 1 through node 5 in the mixture view have a co-regularization relationship with the corresponding node representations of the sub-network Fig. 2(b) and Fig. 2(c). In this way, heterogeneous information may be incorporated into representations of nodes 1 through 5 in the mixture view.

Since the co-regularization is enforced on the representations of the same nodes in different views, the problem becomes how to relate different representations of the same entities. Recall that the node representation for each node in the mixture view encodes all types of edges it is involved in, and a node representation in the split view only encodes one type of edges. Hence, the latter can be viewed as partial information of the expression of the former in a particular interaction environment. Based on the assumption of partial information, we propose a part-based co-regularization; while based on the assumption of the expression, we propose an Auto-Encoder-based co-regularization and a factorization-based co-regularization.

3.4.1 *Part-based Co-regularization*

We can assume that node representations in the split view as partial information of those in the mixture view because the former is learned from a sub-network. Accordingly, the co-regularization can be achieved as follows:

$$\mathcal{R}_{Part} = \sum_r \sum_i \lambda_{(r,i)} ||\boldsymbol{n}_i^m - \boldsymbol{n}_{(r,i)}^s||_2^2, \tag{5}$$

where $\lambda_{(r,i)} \in \mathbb{R}$ is the weight assigned to each node representation in the split view. In this paper, we propose an auto-weighted method presented in the optimization section instead of pre-defining the weight.

3.4.2 *Auto-Encoder-based Co-regularization*

The part-based co-regularization requires that all node representations are in the same space, which may not be reasonable because different sub-networks may have different node and edge types. We thus propose an Auto-Encoder-based co-regularization which models the expression of mixture-view node representations in a particular interaction environment.

The framework of Auto-Encoder [44] is employed in that neural networks can theoretically approximate any kinds of data transformation. In particular, node representations in the mixture view are treated as original data codings and node representations in the split view are treated as latent data codings. But the difference from a typical Auto-Encoder is that original data codings are subject to change via learning from the latent codings. In particular, the objective function of the Auto-Encoder-based co-regularization can be written as follows:

$$\begin{aligned}\mathcal{R}_{AutoEncoder} = \sum_r \Bigg[& \sum_{i \in G_r^s} ||f_{(M,S_r)}(\boldsymbol{n}_i^m) - \boldsymbol{n}_{(r,i)}^s||_2^2 \\ & + \sum_{i \in G_r^s} ||f_{(S_r,M)}(\boldsymbol{n}_{(r,i)}^s) - \boldsymbol{n}_i^m||_2^2 \Bigg],\end{aligned} \tag{6}$$

where $f_{(M,S_r)}(\cdot)$ and $f_{(S_r,M)}(\cdot)$ represent the feedforward neural network of the encoder and the decoder for the sub-network $G_r^s(N_r^s, E_r^s)$ or $G_r^s(N_{r_A}^s \cup N_{r_B}^s, E_r^s)$, respectively.

3.4.3 *Factorization-based Co-regularization*

$\mathcal{R}_{AutoEncoder}$ models non-linear transformation between the representations in the different views. For a more complete study of the proposed mixture-split framework, we further propose a linear-based co-regularization. In particular, we propose to factorize node representations in the split view into the multiplication of node representations in the mixture view and a basis matrix as follows:

$$\mathcal{R}_{Factorization} = \sum_r ||\boldsymbol{N}_r^m \boldsymbol{P}_r - \boldsymbol{N}_r^s||_F^2 \tag{7}$$

,

where $\boldsymbol{N}_r^s \in \mathbb{R}^{|N_r^s| \times d_s}$ is a matrix containing node representations of the sub-network r, $\boldsymbol{N}_r^m \in \mathbb{R}^{|N_r^s| \times d_m}$ is a matrix containing corresponding node representations in the mixture view, and $\boldsymbol{P}_r \in \mathbb{R}^{d_m \times d_s}$is a basis matrix to be estimated. The basis matrix realizes the linear transformation of node representations in the mixture view into the node representations in the split view.

3.5 Joint Learning

The joint learning is to learn node representations via embedding the network structure and performing the co-regularization simultaneously. The overall objective function can be obtained by directly adding each term together denoted as follows:

$$\mathcal{L} = \mathcal{L}^m + \mathcal{R} + \sum_r \mathcal{L}_r^s + \eta \big[||\boldsymbol{N}^m||_F^2 + \sum_r (||\boldsymbol{N}_r^s||_F^2 + ||\boldsymbol{P}_r||_F^2) \big], \tag{8}$$

where $\mathcal{R}$ is a co-regularization term, and can be one of $\mathcal{R}_{Part}$, $\mathcal{R}_{AutoEncoder}$, and $\mathcal{R}_{Factorization}$. In the rest of the paper, the three models with the three co-regularization terms are referred to as MixSp(Part), MixSp(AE), and MixSp(FAC), respectively. $\eta \in \mathbb{R}$ is a regularization coefficient. The Frobenius norm on $\boldsymbol{N}^m$ and $\boldsymbol{N}_r^s$ is to avoid trivial solutions where $\boldsymbol{N}^m$ and $\boldsymbol{N}_r^s$ take large values because large values can easily make the estimated probability of an existing edge, i.e., Eq. (1), close to 1. The regularization on $\boldsymbol{P}_r$ is motivated by regularized matrix factorization. We assume the same η for simplification, and obtain good results in the experiments.

3.6 Optimization

$\mathcal{L}$ is not jointly convex on all the variables. We thus propose to solve each of the variables by alternating optimization. In particular, each variable is alternatingly solved with all the others fixed until convergence.

3.7 Optimization of MixSp(Part)

Note that the loss function for each variable is simplified with all the other variables fixed. For node representations in the mixture view, it is reduced to the following one:

$$\min_{\boldsymbol{N}_i^m} \mathcal{L}^m + \sum_r \sum_i \lambda_{(r,i)} ||\boldsymbol{n}_i^m - \boldsymbol{n}_{(r,i)}^s||_2^2 + \eta ||\boldsymbol{N}^m||_F^2, \tag{9}$$

which can be solved by gradient-based methods, e.g., L-BFGS and gradient descent. We omit the gradient calculation to save space because it is a straightforward procedure.

With respect to the weight $\lambda_{(r,i)}$, inspired by the auto-weighted multi-view learning [45] [46], we propose to learn the weight along with the optimization. In particular, for each node, we first assume an objective with no weights, and take the following general form:

$$\min_{\boldsymbol{n}_i^m} \sum_r ||\boldsymbol{n}_i^m - \boldsymbol{n}_{(r,i)}^s||_2 + \Upsilon(\boldsymbol{n}_i^m), \tag{10}$$

where $\Upsilon(\boldsymbol{n}_i^m)$ contains all the terms related to $\boldsymbol{n}_i^m$ in Eq. (9).

Then, we take the derivative of Eq. (10) w.r.t. $\boldsymbol{n}_i^m$:

$$\sum_r \alpha_{(r,i)} \frac{\partial ||\boldsymbol{n}_i^m - \boldsymbol{n}_{(r,i)}^s||_2^2}{\boldsymbol{n}_i^m} + \frac{\partial \Upsilon(\boldsymbol{n}_i^m)}{\partial \boldsymbol{n}_i^m}, \tag{11}$$

where

$$\alpha_{(r,i)} = \frac{1}{2||\boldsymbol{n}_i^m - \boldsymbol{n}_{(r,i)}^s||_2} \quad (12)$$

Note that when $\alpha_{(r,i)}$ is set to be stationary, gradient-based minimization of Eq. (10) with Eq. (11) as the gradient is the solution to Eq. (9) with respect to $\boldsymbol{n}_i^m$. In fact, $\alpha_{(r,i)}$ is dependent on $\boldsymbol{n}_i^m$. Therefore, we need to solve $\boldsymbol{n}_i^m$ and $\alpha_{(r,i)}$ alternatively in an iterative manner as shown in the following Algorithm 1.

The optimization with respect to node representations in the split view is similar because $\mathcal{L}_r^s$ is similar to $\mathcal{L}^m$, and thus is omitted for space consideration.

3.8 Optimization of MixSp(AE)

The optimization with respect to node representations can also be solved by gradient-based methods. The variables of the co-regularization, i.e., variables in $f_{(M,S_r)}(\cdot)$ and $f_{(S_r,M)}(\cdot)$, are the parameters of feedforward neural networks and they can be solved by back propagation.

3.9 Optimization of MixSp(FAC)

The optimization with respect to node representations can be solved by gradient-based methods. The optimization problem with respect to $\boldsymbol{P}_r$ is reduced to the following one:

$$\min_{\boldsymbol{P}_r} ||\boldsymbol{N}_r^m \boldsymbol{P}_r - \boldsymbol{N}_r^s||_F^2 + \eta||\boldsymbol{P}_r||_F^2 \quad (13)$$

It is easy to see that the optimal $\boldsymbol{P}_r$ can be obtained by setting the derivative of Eq. (13) w.r.t. $\boldsymbol{P}_r$ to zero. After straightforward calculations, the optimal $\boldsymbol{P}_r$ is computed as follows:

$$\boldsymbol{P}_r^* = ((\boldsymbol{N}_r^m)^\top \boldsymbol{N}_r^m + \eta \boldsymbol{I})^{-1} (\boldsymbol{N}_r^m)^\top \boldsymbol{N}_r^s, \quad (14)$$

3.10 Optimization Algorithm

The alternating optimization algorithms for all the three models are similar, and thus a unified one is presented in Algorithm 1. K is the negative ratio used in negative sampling for embedding network structure. Pre-training is an important part of an optimization algorithm as it can initialize a model to a point in parameter space that renders the learning process more effective [47]. In Algorithm 1, pre-training $\boldsymbol{N}^m$ and $\{\boldsymbol{N}_1^s, ..., \boldsymbol{N}_r^s, ...\}$ can ensure the learning of $f_{(M,S_r)}(\cdot)$ and $f_{(S_r,M)}(\cdot)$ or $\{\boldsymbol{P}_1, ..., \boldsymbol{P}_r, ...\}$ non-trivial, and thus make the co-regularization between node representations in different views immediately effective. To this end, the pre-training of $\boldsymbol{N}^m$ is performed by minimizing $\mathcal{L}^m$, i.e., embedding the network in the mixture view, which can be done via gradient descent. The pre-training of each $\boldsymbol{N}_r^s$ is similar. For the learning rate of each iteration in gradient descent, we employ backtracking line search [48] to determine an appropriate one during the training process.

3.10.1 Complexity

The complexity is mainly determined by three procedures, i.e., learning the parameters in the co-regularization, solving $\boldsymbol{N}^m$, and solving $\{\boldsymbol{N}_1^s, ..., \boldsymbol{N}_r^s, ...\}$. The three models have the same level of complexity for solving the node representations. In particular, for $\boldsymbol{N}^m$, the complexity is $O(Kd_m|E^m|)$ if $K|E^m| >= \sum_r |N_r^s|d_s$ or $O(\sum_r |N_r^s|d_m d_s)$ otherwise. For each $\boldsymbol{N}_r^s$, the complexity is $O(Kd_s|E_r^s|)$ if $K|E_r^s| >= |N_r^s|d_m$ or $O(|N_r^s|d_m d_s)$ otherwise.

Algorithm 1: The alternating optimization algorithm

```
Input  : $G(N, E)$, $d_m$, $d_s$, $K$, $\eta$
Output: $\boldsymbol{N}^m$, $\{\boldsymbol{N}_1^s, ..., \boldsymbol{N}_r^s, ...\}$
1  Construct the mixture-split representation of $G(N, E)$;
2  Pre-train $\boldsymbol{N}^m$ and $\{\boldsymbol{N}_1^s, ..., \boldsymbol{N}_r^s, ...\}$;
3  Initiate all $\alpha_{(r,i)}$ by Eq. (12) for MixSp(Part) only;
4  while (not converge) do
5      Fix $\boldsymbol{N}^m$ and $\{\boldsymbol{N}_1^s, ..., \boldsymbol{N}_r^s, ...\}$, solve $f_{(M,S_r)}(\cdot)$ and
        $f_{(S_r,M)}(\cdot)$ with gradient descent or find the
        optimal $\{\boldsymbol{P}_1, ..., \boldsymbol{P}_r, ...\}$ with Eq. (14);
6      Fix all other variables, solve $\boldsymbol{N}^m$ with gradient
        descent;
7      Update all $\alpha_{(r,i)}$ by Eq. (12) for MixSp(Part) only;
8      for ( $r = 1$; $r <= |R|$; $r = r + 1$ ) {
9          Fix all other variables, solve $\boldsymbol{N}_r^s$ with gradient
            descent;
10         }
11     Update all $\alpha_{(r,i)}$ by Eq. (12) for MixSp(Part) only;
12 return $\boldsymbol{N}^m$, $\{\boldsymbol{N}_1^s, ..., \boldsymbol{N}_r^s, ...\}$
```

TABLE 1: Baselines for each type of studied networks.

Network Type	Baselines
Type I	MTNE [6], DMNE [7]
Type II	EOE [8], DMNE [7]
Type III	metapath2vec [10], HEER [11]
All types	Deepwalk [1], LINE [2], node2vec [3]

For MixSp(Part), the complexity of updating η is $O(|N||R|d_m)$ at the worst case where each node participates in every sub-network in the split view. The number of nodes is usually larger than that of nodes. Hence, the complexity is dominated by learning node representations, which is proportional to the number of edges. Since networks are usually sparse in practice [49], the scalability to large-scale networks can be guaranteed.

For MixSp(AE), the number of parameters of the co-regularization depends the architecture of the feedforward neural networks. In the experiments, we use single-layer neural networks for both $f_{(M,S_r)}(\cdot)$ and $f_{(S_r,M)}(\cdot)$ in order to demonstrate that the proposed model does not rely on the benefits brought by deep learning. Accordingly, the number of parameters for each neural network is equal to $d_m d_s$. The complexity for learning the parameters thus can be determined as $O(|N_r^s|d_m d_s)$.

For MixSp(FAC), the optimal $\boldsymbol{P}_r$ is obtained in Eq. (14), which has a complexity of $O(|N_r^s|d_m^2)$.

3.10.2 Convergence

Algorithm 1 is essentially a block-wise coordinate descent algorithm [50] with $\boldsymbol{N}^m$, $\{\boldsymbol{N}_1^s, ..., \boldsymbol{N}_r^s, ...\}$, and variables involved in the co-regularization as block variables. So convergence can be guaranteed based on the general proof of convergence for block-wise coordinate descent. Also, in the experiments, we observe Algorithm 1 converges fast in terms of the iterations in the "while-do" loop.

TABLE 2: Statistics of Type I, Type II, and Type III networks

Network Type	Dataset	DBLP			Flickr		
Type I	Network	Coauthor	Citation	KNN	Tag	Group	Location
	#Edges	19265	120760	421330	169954	74803	13916
	#Nodes	6482 authors			4418 photos		

Network Type	Dataset	DBLP			BlogCatalog		
Type II	Edge type	Coauthor	Authorship	Word Co-occurrence	Friends	Authorship	Word Co-occurrence
	#Edges	19265	1265023	1146128	28406	344942	885207
	#Nodes	6482 authors, 8302 words			5009 users, 8391 words		

Network Type	Dataset	DBLP				
Type III	Edge type	Coauthor	Authorship	Paper Citation	Paper-Conference	Author-Conference
	#Edges	19265	29175	18325	9872	15676
	#Nodes	6482 authors, 9872 papers, 20 conferences				

4 Empirical Evaluation

4.1 Experiment Settings

The performance of the proposed MixSp models is evaluated on multi-label node classification and link prediction tasks. Since MixSp models are general-purpose, we thus study several different types of heterogeneous information networks. In particular, the three types of HINs mentioned in the related work are all studied, i.e., Type I, Type II, and Type III networks. For all the three types of networks, the respective baselines are summarized in TABLE I where each baseline has been introduced in the related work. Note that homogeneous network embedding methods, i.e., Deepwalk [1], LINE [2], and node2vec [3], are applicable to all types of HINs when the networks are treated as homogeneous.

For the implementation of Algorithm 1, dimensions d_m and d_s are set as 128 which is also used in baselines, K for the negative sampling is set as 5 like in LINE, regularization coefficient η is set as 1, and the relative loss for determining the convergence of each gradient descent is set as 0.0001. In the experiments, Algorithm 1 is implemented in Java and run on an Intel Genuine Intel(R) CPU @2.60GHZ 2.60GHZ server with 32 GB RAM.

4.2 Datasets

The experiments are conducted on three real-world datasets which are described as follows:

- DBLP [51]: This dataset contains bibliographic information of major computer science publications from the dblp computer science bibliography. The dataset can be used to construct all the aforementioned three types of heterogeneous information networks based on many kinds of relationships, e.g., co-authorship, author-paper, paper-abstract, paper-venue relationships, etc.
- BlogCatelog [52]: This dataset has information about blogs and friendships of users in BlogCatalog, an online blogging collaboration application. The network consisting of users and words as nodes and user friendships, user-word authorships, and word co-occurrences as edges belongs to the Type II network.
- Flickr [53]: This dataset is about photo information obtained from Flickr, an online photo sharing application. Each photo has tag, group, location and label information. Relationships can be established among photos as the sharing of the same entities, e.g., tag, group, or location. The network consisting of only photos as nodes but multiple types of edges belongs to the Type I network.

The statistics of networks constructed from the three datasets are presented in TABLE II.

4.2.1 *Type I Network*

To construct the network from the DBLP dataset, papers are selected from popular conferences of four fields, which are SIGMOD, VLDB, ICDE, EDBT, and PODS for Database, KDD, ICDM, SDM, and PAKDD for Data Mining, ICML, NIPS, AAAI, IJCAI and ECML for Machine Learning, and SIGIR, WSDM, WWW, CIKM, and ECIR for Information Retrieval. The publication time is from 2000 to 2009. We select active authors with more than two papers during the time period. For the KNN sub-network, the similarity among authors is quantified by cosine similarity of term frequency of keywords in their papers' abstract. Keywords with overall frequency less than 7 are omitted. The first 1% authors are set as the number of neighbors.

For the photo network of Flickr, the edges are established between photos sharing the same tag, group and location, respectively.

It is worth mentioning that for both of the two networks, not all the nodes are involved in every sub-network. For example, it is difficult to ensure that the set of authors involved in co-authorships have citations to each other, and vice versa. The scenario can be viewed as the cold-start problem [54] for each sub-network in which some nodes may not have edges to other nodes due to some reasons in practice. Accordingly, multi-view methods, e.g., MVE [33] which requires each node participates in every view, may not be applicable.

4.2.2 *Type II Network*

The same set of papers selected for the Type I network of DBLP is used to construct the Type II network. The sliding window for constructing word co-occurrences in sentences of paper abstract is set as 5.

For the BlogCatalog data, the users are selected from four major categories including Art, Computers, Music, and Photography. And their blogs related to the four categories are used to construct the word sub-network. Words with frequency less than 8 are filtered out.

TABLE 3: Multi-label classification for Type I, Type II, and Type III networks.

Network Type	Dataset	Metric	Deepwalk	LINE(1st)	LINE(2nd)	node2vec	MTNE	DMNE	MixSp(Part)	MixSp(AE)	MixSp(FAC)
Type I	DBLP	Micro-F1(%)	80.9	81.9	81.1	81.8	84.3	81.9	**84.4**	84.2	84.1
		Macro-F1(%)	79.8	80.9	80.3	80.9	83.5	81.2	**83.7**	83.5	83.4
	Flickr	Micro-F1(%)	61.0	61.5	60.0	61.0	62.9	59.1	**63.2**	**63.2**	**63.2**
		Macro-F1(%)	13.5	15.6	12.4	13.8	23.3	12.1	25.7	25.7	**25.8**

Network Type	Dataset	Metric	Deepwalk	LINE(1st)	LINE(2nd)	node2vec	EOE	DMNE	MixSp(Part)	MixSp(AE)	MixSp(FAC)
Type II	DBLP	Micro-F1(%)	74.5	82.4	72.4	76.0	79.0	76.6	**84.9**	83.9	84.0
		Macro-F1(%)	72.9	81.5	69.6	73.5	72.2	71.8	**84.3**	82.5	82.7
	Blog	Micro-F1(%)	66.8	71.6	59.2	67.2	80.53	79.9	81.6	**82.3**	81.9
		Macro-F1(%)	53.9	63.7	27.4	55.2	75.81	78.6	77.9	**78.8**	78.4

Network Type	Dataset	Metric	Deepwalk	LINE(1st)	LINE(2nd)	node2vec	metapath2vec	HEER	MixSp(Part)	MixSp(AE)	MixSp(FAC)
Type III	DBLP	Micro-F1(%)	95.2	94.4	76.5	93.1	96.9	96.6	**98.9**	**98.9**	**98.9**
		Macro-F1(%)	95.0	94.1	75.0	92. 7	96.7	96.3	**98.9**	**98.9**	**98.9**

4.2.3 *Type III Network*

From the previously selected papers, we construct a network with three types of nodes which are author, paper, and conference venue, and five types of edges which are co-authorships, authorships, paper citations, paper-conference relationships, and author-conference relationships.

4.3 Multi-label Node Classification

In multi-label classification, each data point is assigned with more than one label. For the DBLP network, the research fields are employed as the labels. For the Flickr network, there are 99 labels indicated in the dataset. For the Blog-Catalog network, the four categories are used as labels. We employ the SVM with polynomial kernel implemented in Meka [55] as the classifier, and use 5-fold cross validation as the evaluation method, and obtain the Micro-F1 and Macro-F1 scores as performance metrics. For MixSp, node representations in the mixture view are concatenated to node representations in the split view to perform the task. The experiment results are presented in TABLE III.

It shows that the best performing model is always one of the variants of MixSp. In fact, MixSp(Part) is always better than all of the baselines, while MixSp(AE) and MixSp(FAC) are almost always better except on the Type I DBLP network. For the Type I network, the reason behind superior performance to Deepwalk, LINE and node2vec may be because MixSp models can obtain more complete information. In particular, each pair of nodes may have multiple types of edges and each type of edges may be established due to different reasons. For example, the three type of edges in the Flickr network are based on the sharing of different concepts, i.e., tag, group, and location. Merging different types of edges into a weighted one would cause information loss. Note that Deepwalk and LINE cannot be applied to each sub-network and then concatenate representations with respect to each sub-network to avoid the information loss due to the cold-start problem mentioned in the last paragraph of Section 4.2.1.

For a fair comparison, we have evaluated the performance of only node representations in the mixture view and also obtain better performance (omitted due to space limit) than Deepwalk, LINE and node2vec, which has actually verified that the node representations in the mixture view can be improved by the node representations in the split view through the co-regularization.

DMNE and MTNE can avoid the information loss but they cannot utilize the information of the mixture network. Note that MixSp models are similar to DMNE and MTNE in the sense that all the sub-networks in the split view are jointly embedded via co-regularization on the node representations. But the co-regularization in MixSp modes involve additional information other than the sub-networks, i.e., the mixture network. The mixture network has been demonstrated to be effective for the task by the performance of Deepwalk and LINE. Intuitively, the network in the mixture view represents a different view of the given HIN from all the sub-networks in the split view, and hence more comprehensive information can be obtained when both two views are embedded, which actually has been the motivation for multi-view learning in image processing tasks where multiple views are constructed from a singe image [12]. In our case, more comprehensive information can be obtained because the two different views have different network structures, and thus can produce two different sets of node representations.

To demonstrate this point, we visualize the concatenated node representations in the split view and the node representations in the mixture view learned by MixSp(Part) using t-SNE [56] in Fig. 3(a) and Fig. 3(b), respectively. Here, the field of a researcher is determined as the one where he/she published the most papers. Since both of them are faithful to the network structure (i.e., researchers of the same fields are distributed together), the combination of them can perform better than any one of them. Besides, the claim that node representations in the mixture view would be improved by node representations in the split view is verified by the better layout of Fig. 3(b) than that of Fig. 3(c) since the network in the mixture view is embedded by the technique proposed in LINE(1st).

For the Type II and the Type III networks, the major reason behind of the superior performance of MixSp models is still the utilization of two views of a single network.

Note that the performance on the Type III DBLP network is considerably better than that on both the Type I and the Type II DBLP networks. This is because the labels, i.e.,

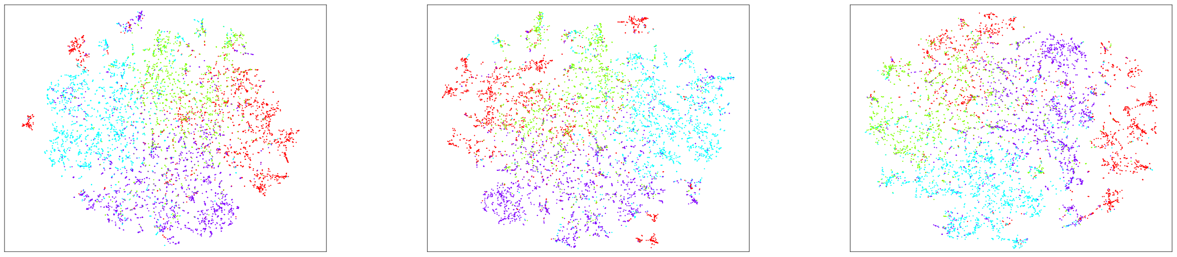

(a) Split View (b) Mixture View (c) LINE(1st)

Fig. 3: Visualization of node representations learned from the Type I DBLP network where colors indicate research fields, i.e., red for Machine Learning, green for Data Mining, light blue for Database, and dark blue for Information Retrieval, respectively.

TABLE 4: Co-authorship Prediction for Type I, Type II, and Type III DBLP networks.

AUC(%)	Deepwalk	LINE(1st)	LINE(2nd)	node2vec	MTNE	DMNE	MixSp(Part)	MixSp(AE)	MixSp(FAC)
Type I	74.07	64.05	66.25	73.26	78.92	61.15	**80.27**	76.29	79.12

AUC(%)	Deepwalk	LINE(1st)	LINE(2nd)	node2vec	EOE	DMNE	MixSp(Part)	MixSp(AE)	MixSp(FAC)
Type II	66.72	60.73	62.75	68.15	75.62	55.32	75.37	75.38	**77.74**

AUC(%)	Deepwalk	LINE(1st)	LINE(2nd)	node2vec	metapath2vec	HEER	MixSp(Part)	MixSp(AE)	MixSp(FAC)
Type III	69.30	63.91	65.89	72.34	49.36	75.8	**79.58**	76.26	75.53

research fields, are defined on the conferences. When the label-related information are explicitly utilized, the performance of the classification can be significantly improved. For LINE(2nd) which dose not enjoy this benefit, the reason may be that LINE(2nd) fails to consider the first-order edges but only embeds the second-order edges.

4.4 Link Prediction

Link prediction refers to inferring new interactions between network nodes, and is commonly performed by measuring similarities of nodes because nodes sharing similar characteristics are more likely to interact [57]. Here, we use the inner product of node representations normalized by sigmoid function to measure the similarity. We predict co-authorships occurring after the time horizon used in the construction of the networks. Actual co-authorships are used as the positive test links. Besides, the same number of non-existing co-authorships are randomly sampled as negative test links for the evaluation purpose. Unlike the multi-label classification, only the node representations corresponding to the co-authorship sub-network are employed to perform the task due to the reason mentioned in the introduction. The performance measured by area under the curve (AUC) is shown in TABLE IV.

It shows that the best performing model is always one of the variants of MixSp. For the Type I network, we report the performance of Deepwalk, LINE and node2vec when they only embed the co-authorship sub-network because they perform even worse when they embed all the three sub-networks as a mixture network. MixSp outperforms Deepwalk, LINE and node2vec because the latter only utilize the previous co-authorships to predict future co-authorships. But it is intuitive that the citation sub-network and the KNN sub-network can provide references to the co-authorship sub-network. For MixSp models, we have also evaluated the node representations corresponding to the mixture network but the performance is inferior to the subnetwork-specific node representations. The experiments collectively verify that other types of edges can provide useful references but should not play a role as important as the target edges. The reason behind the poor performance of DMNE which also implicitly utilizes all other types of edges for the prediction of co-authorships is beyond this paper. But we find that DMNE is not evaluated on the link prediction task in its paper.

For the Type II network, MixSp outperforms Deepwalk, LINE and node2vec because the latter does not distinguish the semantic meanings of different nodes. In particular, for the DBLP network, the result of not distinguishing authors and words is that authors connected to similar words would have large similarities but it is not necessary for two authors have similar research interests to co-author a paper.

For the Type III network, the poor performance of metapath2vec may be because the conference information is explicitly embedded into the author representations, but many authors attending the same conferences do not necessarily co-author a paper.

4.5 Discussions

The DBLP dataset is studied in terms of the three types of HINs, but the three networks have different information, e.g., the Type III network has conference information which is directly related to the labels. As a result, the performance of the multi-label classification on the Type III network is significantly better than that on the other two types of networks. The superior performance on the Type I network to that on the Type II network is because besides the co-authorship and paper abstract information, the author citation information is further considered. Hence, it seems to be better to utilize more relevant information for the classification task. For the link prediction task, however, the experiments do not show more information always brings better performance. This may be due to the specificity of the link prediction task which is specific to the type of edges of interest, which may explain why MixSp models do no consistently outperform the network-specific models (e.g., EOE) on the link prediction tasks. Accordingly, one open issue can be how to select useful information for the embedding methods to perform the link prediction task.

Among the three MixSp models, MixSp(Part) obtains the best performance at most times. This may be because enforcing all the node representations in the same space is more effective for the co-regularization. But this treatment may be less appropriate in some scenarios where node types are different, e.g., MixSp(Part) obtains the worst performance for the multi-label classification task on the Type II network of BlogCatalog, because different node types have different semantic meanings.

4.6 Parameter Sensitivity

Previous experiments are conducted with the regularization coefficient (η) and the dimension (d_m and d_s) of node representations fixed as 1 and 128, respectively. This section thus studies the sensitivity of the performance with respect to these two hyper-parameters. In particular, we study the coefficient in the range $\{0.01, 0.1, 1, 10, 100\}$ with the dimension fixed as 128, and study the dimension in the range $\{32, 64, 128, 256, 512\}$ with the coefficient fixed as 1. We only present results obtained on the co-authorship prediction for the Type I DBLP network in Fig. 4(a) and Fig. 4(b) because other experiments show similar patterns. For the regularization coefficient, it shows that the value should not be very large (e.g., 100).

As for the dimension, it shows that the performance is stable within the studied range. It also shows that the performance decreases when the dimension increases. After double-checking the papers of baselines, such LINE and Deepwalk, we find that it is quite common for embedding models that the performance of subsequent data mining tasks goes down when the dimension goes up. We think this phenomenon is due to the curse of dimensionality. Particularly, we tend to think that the curse of dimensionality is not on the network embedding models, but on the models for classification and link prediction tasks.

4.7 Convergence Analysis

In this section, we empirically study the convergence of the Algorithm 1. Specifically, we study the loss function and the performance of the algorithm on data mining tasks with respect to the number of iterations in the while-do loop. We only present the results of multi-label classification obtained by MixSp(Part) on the Type II networks in Fig. 5 because other experiments show similar results. In the experiments, each iteration takes about 1623 seconds and 5055 seconds on average for the Type II BlogCatalog and DBLP network, respectively. Fig. 5 shows that the algorithm can usually converge to stable performance after about 15 iterations.

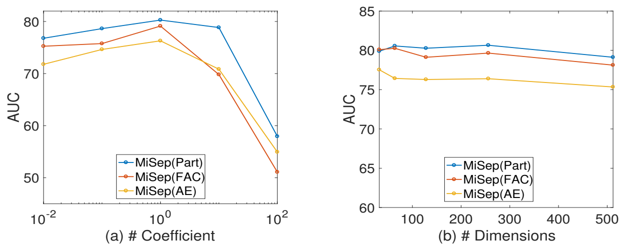


Fig. 4: Parameter sensitivity.

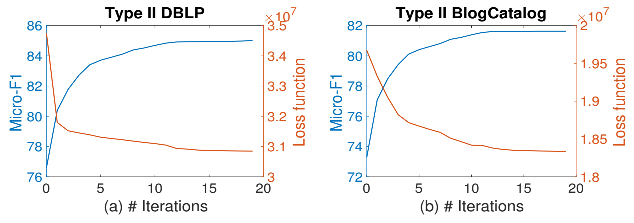


Fig. 5: Convergence analysis.

5 Conclusion and Future Work

In this paper, we propose MixSp, a framework for embedding HINs with arbitrary number of node and edge types. For any given HIN, MixSp constructs a mixture-split representation where the network in the mixture view is treated as a homogeneous network, and each sub-network in the split view only consists of one type of edges with the corresponding nodes. Node representations in each view not only encode the structure of the respective network(s), but also are regularized by the corresponding node representations in the other view. The three models are just three instantiations of the proposed framework, and variants can be developed when different techniques are employed for the network structure embedding and different mechanisms are designed for the co-regularization.

Acknowledgments

This work was partially supported by JST KAKENHI 19H01114 and JST-AIP JPMJCR19U4, and NSF under grants III-1526499, III-1763325, III-1909323, and CNS-1930941, and NSF of Guangdong Province through grant 2017A030313339, HK RGC Collaborative Research Fund (CRF), Project No. C5026-18G, and HK RGC Collaborative Research Fund (CRF), Project No. C6030-18G, and HK RGC General Research Fund (GRF), PolyU 152199/17E.

References

[1] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: online learning of social representations," in *SIGKDD*. ACM, 2014, pp. 701–710.
[2] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *The Web Conference*. International World Wide Web Conferences Steering Committee, 2015, pp. 1067–1077.
[3] A. Grover and J. Leskovec, "node2vec: scalable feature learning for networks," in *SIGKDD*. ACM, 2016, pp. 855–864.
[4] J. Han, "Mining heterogeneous information networks by exploring the power of links," in *DS*. Springer, 2009, pp. 13–30.
[5] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu, "Pathsim: Meta path-based top-k similarity search in heterogeneous information networks," *Proceedings of the VLDB Endowment*, vol. 4, no. 11, pp. 992–1003, 2011.
[6] L. Xu, X. Wei, J. Cao, and S. Y. Philip, "Multi-task network embedding," *International Journal of Data Science and Analytics*, vol. 8, no. 2, pp. 183–198, 2019.

[7] J. Ni, S. Chang, X. Liu, W. Cheng, H. Chen, D. Xu, and X. Zhang, "Co-regularized deep multi-network embedding," in *The Web Conference*. International World Wide Web Conferences Steering Committee, 2018, pp. 469–478.

[8] L. Xu, X. Wei, J. Cao, and P. S. Yu, "Embedding of embedding (eoe): Joint embedding for coupled heterogeneous networks," in *WSDM*. ACM, 2017, pp. 741–749.

[9] J. Tang, M. Qu, and Q. Mei, "Pte: Predictive text embedding through large-scale heterogeneous text networks," in *SIGKDD*. ACM, 2015, pp. 1165–1174.

[10] Y. Dong, N. V. Chawla, and A. Swami, "metapath2vec: Scalable representation learning for heterogeneous networks," in *SIGKDD*. ACM, 2017, pp. 135–144.

[11] Y. Shi, Q. Zhu, F. Guo, C. Zhang, and J. Han, "Easing embedding learning by comprehensive transcription of heterogeneous information networks," in *SIGKDD*. ACM, 2018, pp. 2190–2199.

[12] X. Cao, C. Zhang, H. Fu, S. Liu, and H. Zhang, "Diversity-induced multi-view subspace clustering," in *CVPR*. IEEE, 2015, pp. 586–594.

[13] L. F. Ribeiro, P. H. Saverese, and D. R. Figueiredo, "struc2vec: learning node representations from structural identity," in *SIGKDD*. ACM, 2017, pp. 385–394.

[14] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[15] C. Yang, Z. Liu, D. Zhao, M. Sun, and E. Y. Chang, "Network representation learning with rich text information." in *IJCAI*, 2015, pp. 2111–2117.

[16] S. Cao, W. Lu, and Q. Xu, "Grarep: learning graph representations with global structural information," in *CIKM*. ACM, 2015, pp. 891–900.

[17] J. Qiu, Y. Dong, H. Ma, J. Li, K. Wang, and J. Tang, "Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec," in *WSDM*. ACM, 2018, pp. 459–467.

[18] X. Wang, P. Cui, J. Wang, J. Pei, W. Zhu, and S. Yang, "Community preserving network embedding," in *AAAI*, 2017, pp. 203–209.

[19] L. Xu, X. Wei, J. Cao, and S. Y. Philip, "Multiple social role embedding," in *DSAA*. IEEE, 2017, pp. 581–589.

[20] L. Xu, J. Cao, X. Wei, and P. Yu, "Network embedding via coupled kernelized multi-dimensional array factorization," *IEEE Transactions on Knowledge and Data Engineering*, 2019.

[21] Y. Gu, Y. Sun, Y. Li, and Y. Yang, "Rare: social rank regulated large-scale network embedding," in *The Web Conference*. International World Wide Web Conferences Steering Committee, 2018, pp. 359–368.

[22] S. Cao, W. Lu, and Q. Xu, "Deep neural networks for learning graph representations," in *AAAI*, 2016, pp. 1145–1152.

[23] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *SIGKDD*. ACM, 2016, pp. 1225–1234.

[24] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *NeurIPS*, 2014, pp. 2672–2680.

[25] H. Wang, J. Wang, J. Wang, M. Zhao, W. Zhang, F. Zhang, X. Xie, and M. Guo, "Graphgan: Graph representation learning with generative adversarial nets," in *AAAI*, 2017, pp. 2508–2515.

[26] J. Zhang, L. Cui, and Y. Fu, "Latte: application oriented network embedding," *arXiv preprint arXiv:1711.11466*, 2017.

[27] R. A. Rossi, R. Zhou, and N. K. Ahmed, "Deep inductive network representation learning," in *The Web Conference*. International World Wide Web Conferences Steering Committee, 2018, pp. 953–960.

[28] L. Xu, X. Wei, J. Cao, and S. Y. Philip, "Interaction content aware network embedding via co-embedding of nodes and edges," in *PAKDD*. Springer, 2018, pp. 183–195.

[29] L. Xu, X. Wei, J. Cao, and P. S. Yu, "On exploring semantic meanings of links for embedding social networks," in *The Web Conference*. International World Wide Web Conferences Steering Committee, 2018, pp. 479–488.

[30] X. Wei, L. Xu, B. Cao, and P. S. Yu, "Cross view link prediction by learning noise-resilient representation consensus," in *The Web Conference*. International World Wide Web Conferences Steering Committee, 2017, pp. 1611–1619.

[31] S. Wasserman and K. Faust, *Social network analysis: Methods and applications*. Cambridge University Press, 1994, vol. 8.

[32] S. Parameswaran and K. Q. Weinberger, "Large margin multi-task metric learning," in *NeurIPS*, 2010, pp. 1867–1875.

[33] J. T. Meng, J. Shang, X. Ren, M. Zhang, and J. Han, "An attention-based collaboration framework for multi-view network representation learning," in *CIKM*. ACM, 2017, pp. 1767–1776.

[34] Y. Shi, F. Han, X. He, X. He, C. Yang, J. Luo, and J. Han, "mvn2vec: preservation and collaboration in multi-view network embedding," *arXiv preprint arXiv:1801.06597*, 2018.

[35] Y. Sun, N. Bu, T.-Y. Hsieh, and V. Honavar, "Multi-view network embedding via graph factorization clustering and co-regularized multi-view agreement," *arXiv preprint arXiv:1811.02616*, 2018.

[36] J. Li, C. Chen, H. Tong, and H. Liu, "Multi-layered network embedding," in *SDM*. SIAM, 2018, pp. 684–692.

[37] T.-y. Fu, W.-C. Lee, and Z. Lei, "Hin2vec: explore meta-paths in heterogeneous information networks for representation learning," in *CIKM*. ACM, 2017, pp. 1797–1806.

[38] J. Shang, M. Qu, J. Liu, L. M. Kaplan, J. Han, and J. Peng, "Meta-path guided embedding for similarity search in large-scale heterogeneous information networks," *arXiv preprint arXiv:1610.09769*, 2016.

[39] T. Chen and Y. Sun, "Task-guided and path-augmented heterogeneous network embedding for author identification," in *WSDM*. ACM, 2017, pp. 295–304.

[40] S. Chang, W. Han, J. Tang, G.-J. Qi, C. C. Aggarwal, and T. S. Huang, "Heterogeneous network embedding via deep architectures," in *SIGKDD*. ACM, 2015, pp. 119–128.

[41] J. Zhang, C. Xia, C. Zhang, L. Cui, Y. Fu, and S. Y. Philip, "Bl-mne: emerging heterogeneous social network embedding through broad learning with aligned autoencoder," in *ICDM*. IEEE, 2017, pp. 605–614.

[42] Y. Ma, Z. Ren, Z. Jiang, J. Tang, and D. Yin, "Multi-dimensional network embedding with hierarchical structure," in *WSDM*. ACM, 2018, pp. 387–395.

[43] Y. Shi, H. Gui, Q. Zhu, L. Kaplan, and J. Han, "Aspem: embedding learning by aspects in heterogeneous information networks," in *SDM*. SIAM, 2018, pp. 144–152.

[44] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[45] F. Nie, J. Li, and X. Li, "Parameter-free auto-weighted multiple graph learning: a framework for multi-view clustering and semi-supervised classification," in *IJCAI*, 2016, pp. 1881–1887.

[46] G. Ma, C.-T. Lu, L. He, S. Y. Philip, and A. B. Ragin, "Multi-view graph embedding with hub detection for brain network analysis," in *ICDM*. IEEE, 2017, pp. 967–972.

[47] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *NeurIPS*, 2007, pp. 153–160.

[48] L. Armijo, "Minimization of functions having lipschitz continuous first partial derivatives," *Pacific Journal of Mathematics*, vol. 16, no. 1, pp. 1–3, 1966.

[49] M. E. Newman, "The structure and function of complex networks," *SIAM Review*, vol. 45, no. 2, pp. 167–256, 2003.

[50] P. Tseng, "Convergence of a block coordinate descent method for nondifferentiable minimization," *Journal of optimization theory and applications*, vol. 109, no. 3, pp. 475–494, 2001.

[51] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su, "Arnetminer: extraction and mining of academic social networks," in *SIGKDD*. ACM, 2008, pp. 990–998.

[52] X. Wang, L. Tang, H. Gao, and H. Liu, "Discovering overlapping groups in social media," in *ICDM*. IEEE, 2010, pp. 569–578.

[53] J. McAuley and J. Leskovec, "Image labeling on a network: using social-network metadata for image classification," in *ECCV*. Springer, 2012, pp. 828–841.

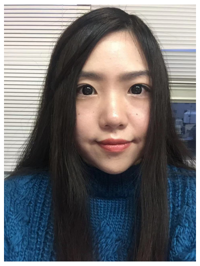
[54] V. Leroy, B. B. Cambazoglu, and F. Bonchi, "Cold start link prediction," in *SIGKDD*. ACM, 2010, pp. 393–402.

[55] J. Read, P. Reutemann, B. Pfahringer, and G. Holmes, "Meka: a multi-label/multi-target extension to weka," *Journal of Machine Learning Research*, vol. 17, no. 1, pp. 667–671, 2016.

[56] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. 1, pp. 2579–2605, 2008.

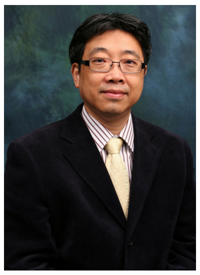
[57] D. Liben-Nowell and J. Kleinberg, "The link-prediction problem for social networks," *Journal of the American Society for Information Science and Technology*, vol. 58, no. 7, pp. 1019–1031, 2007.

Linchuan Xu received the B.E. degree in information engineering from Beijing University of Posts and Telecommunications in 2013, and Ph.D. degree from Department of Computing of the Hong Kong Polytechnic University, in 2018. He is currently a post-doctoral researcher of Department of Mathematical Informatics, Graduate School of Information Science and Technology at the University of Tokyo. His current research interests include data mining and big data with emphasis on network data analytics and medical data analytics.



Jing Wang received the M.Sc. degree in multimedia information technology from the City University of Hong Kong, Hong Kong, in 2012, and the Ph.D. from the Faculty of Science and Technology, Bournemouth University, U.K, in 2018. Currently, she is a post-doctoral researcher in the University of Tokyo, Japan. Her current research interests include machine learning, computer vision, and data mining.

Lifang He received the B.S. degree in Computational Mathematics from Northwest Normal University in 2009, and the Ph.D. degree in School of Computer Science and Engineering, South China University of Technology in 2014. She is currently an Assistant Professor in the Department of Computer Science and Engineering at Lehigh University. Before her current position, Dr. He worked as a postdoctoral researcher in the Department of Biostatistics and Epidemiology at the University of Pennsylvania. Her current research interests include machine learning, data mining, tensor analysis, with major applications in biomedical data and neuroscience.



Jiannong Cao received the B.S. degree in computer science from Nanjing University, China, in 1982, and the M.Sc. and Ph.D. degrees in computer science from Washington State University, USA, in 1986 and 1990 respectively. He is currently a Chair Professor of Department of Computing at The Hong Kong Polytechnic University, Hong Kong. His research interests include parallel and distributed computing, wireless networks and mobile computing, big data and cloud computing, pervasive computing, and fault tolerant computing. He has co-authored 5 books in Mobile Computing and Wireless Sensor Networks, co-edited 9 books, and published over 500 papers in major international journals and conference proceedings. He is a fellow of IEEE, a distinguished member of ACM, a senior member of China Computer Federation (CCF).

Xiaokai Wei received the B.S. degree from Beijing University of Posts and Telecommunications and Ph.D. degree from University of Illinois at Chicago, both in computer science. He joined Facebook Inc. as a research scientist in 2016. His main research areas are data mining and machine learning, especially feature selection and social network mining. He has published more than 20 papers in refereed journals and conferences, such as WWW, WSDM, AAAI, AISTATS, ICDM, SDM, ECML/PKDD.

Philip S. Yu received the B.S. Degree in E.E. from National Taiwan University, the M.S. and Ph.D. degrees in E.E. from Stanford University, and the M.B.A. degree from New York University. He is a Distinguished Professor in Computer Science at the University of Illinois at Chicago and also holds the Wexler Chair in Information Technology. Before joining UIC, Dr. Yu was with IBM, where he was manager of the Software Tools and Techniques department at the Watson Research Center. His research interest is on big data, including data mining, data stream, database and privacy. He has published more than 1,000 papers in refereed journals and conferences. He holds or has applied for more than 300 US patents. Dr. Yu is a Fellow of the ACM and the IEEE. Dr. Yu is the recipient of ACM SIGKDD 2016 Innovation Award for his influential research and scientific contributions on mining, fusion and anonymization of big data, the IEEE Computer Society 2013 Technical Achievement Award for “pioneering and fundamentally innovative contributions to the scalable indexing, querying, searching, mining and anonymization of big data”, and the Research Contributions Award from IEEE Intl. Conference on Data Mining (ICDM) in 2003 for his pioneering contributions to thefield of data mining. He also received the ICDM 2013 10-year Highest-Impact Paper Award, and the EDBT Test of Time Award (2014). He was the Editor-in-Chiefs of ACM Transactions on Knowledge Discovery from Data (2011-2017) and IEEE Transactions on Knowledge and Data Engineering (2001-2004).

Kenji Yamanishi received the M.E. and Dr.Eng. degrees from The University of Tokyo, Japan, in 1987 and 1992, respectively. He was with NEC Corporation from 1987 to 2008. He was a Visiting Scientist with the NEC Research Institute, USA, from 1992 to 1995. Since 2009, he has been leading the Information-Theoretic Machine Learning and Data Mining Group, The University of Tokyo, where he is currently a Professor with the Department of Mathematical Informatics. His interests include information-theoretic machine learning, statistical model selection, and data mining with applications to healthcare and traffic mining.