12-2020

Co-embedding attributed networks with external knowledge

Pei-chi LO
Singapore Management University, pclo.2017@phdcs.smu.edu.sg

Ee-peng LIM
Singapore Management University, eplim@smu.edu.sg

Co-Embedding Attributed Networks with External Knowledge

Pei-Chi Lo
School of Information System
Singapore Management University
Singapore, Singapore
pclo.2017@phdcs.smu.edu.sg

Ee-Peng Lim
School of Information System
Singapore Management University
Singapore, Singapore
eplim@smu.edu.sg

***Abstract*—Attributed network embedding aims to learn representations of nodes and their attributes in a low-dimensional space that preserves their semantics. The existing embedding models, however, consider node connectivity and node attributes only while ignoring external knowledge that can enhance node representations for downstream applications. In this paper, we propose a set of new VAE-based embedding models called External Knowledge-Aware Co-Embedding Attributed Network (ECAN) Embeddings to incorporate associations among attributes from relevant external knowledge. Such external knowledge can be extracted from text corpus and knowledge graphs. We use multi-VAE structures to model the attribute associations. To cope with joint encoding of attribute semantics from different sources, we introduce a mixed model variant which has a two-layer encoder structure. Our experiments on three real-world datasets show that ECAN out-performs previous approaches in both node classification and link prediction tasks.**

***Index Terms*—Attributed Networks, Network Embedding, Knowledge Graph**

I. Introduction

Motivation. Network embedding has been an important research topic in recent years due to the existence of massive amount of web and social network data as well as many applications on these networks, e.g., link prediction, recommendation, node classification, etc.. Many of these networks are attributed networks, a type of networks consisting of both nodes and their attributes. For example, in a citation network, nodes are publications and node attributes include paper title, author, and abstract. The node attributes themselves may also be semantically associated with one another. E.g., two keywords (e.g., "AI" and "deep learning") in abstract can be semantically related if one has the background research knowledge. There are several efforts to incorporate node attributes into network embeddings such as [1] and [2] but they have not considered external attribute semantics in their models.

Increasingly, attribute semantics can be found in publicly available external knowledge sources. Examples of such external knowledge sources include Wikipedia, text corpuses, WordNet, etc.. From those sources, we can derive attribute semantics that come in the form of associations between words and entities for learning better embedding representations of both nodes and attributes.

Consider attributed network example with publications as nodes where v_1 and v_2 are two publication nodes with title attributes, "Mining Authority Nodes from Hyperlinks in WWW" and "The PageRank Citation Ranking: Bringing Order to the Web". The two publications share similar topic but they have little overlap between their titles. We consider both entities and words in these publication titles. With external knowledge, we can determine that the entities that "Web" and "WWW" in the titles are highly similar, and another two entities "Authority" and "Pagerank" are similar. At the *word* level, "hyperlinks" and "web" are semantically similar. These semantic knowledge should bring the embeddings of v_1 and v_2 closer to each other.

Research Objectives. In this paper, we therefore focus on incorporating external attribute knowledge into attributed network embedding. We specifically study this for the variational autoencoder-based (VAE-based) models which has seen much development in recent research. One has to overcome a few technical challenges in considering attribute knowledge, namely: (a) extracting attribute semantics from external knowledge sources; (b) designing new VAE architecture(s) to incorporate different attribute semantics into both node and attribute embeddings; and (c) evaluation of the proposed VAE.

Contribution. We propose a novel framework and several variants of External Knowledge-Aware Co-Embedding Attribute Network (ECAN) embedding models that incorporate external attribute semantics. We introduce two ways to measure the strength of attribute associations, i.e., (i) contextual distance between words in a text corpus, and (ii) similarity between entities in a knowledge graph. Our proposed ECAN models learn both entity and attribute representations in the same embedding space with a multi-VAE architecture. Specifically, we introduce a mixed model variant to merge the encoding of attribute semantics from different knowledge sources. Finally, We conduct extensive experiments on three real-world datasets and show that ECAN models, especially the mixed model variants, significantly outperform the state-of-the-art model.

II. Related Work

Heterogeneous Information Network (HINs) embeddings aim to learn low-dimensional representations of nodes that

978-1-7281-6251-5/20/$31.00 ©2020 IEEE

preserve the network structure [3], [4]. Beyond network structures, one can improve the node embeddings by considering node attributes, node labels and text content associated with the nodes [1], [5]–[7]. For example, TriDNR considers both node content and node label in the learning of node embeddings [8]. It jointly optimizes a skip-gram model for network structure, and a coupled neural network which maps the node content and node representation from skip-gram to the node label. ANRL utilizes an auto-encoder to learn node embedding that preserves both the network structure and node attribute proximity [9]. Nevertheless, even when auxiliary node information (e.g., label and attributes) are modeled by the above embedding approaches, they have yet considered: (1) uncertainty of node representation, and (2) attribute representations. These two properties are important as the former avoids over-fitting and enables generative process, while the latter provides a common representation to compare nodes and attributes. In this paper, we utilize variational auto-encoder to address these shortcomings.

VAE, a popular representation learning for text/image generation [10]–[12], has shown to outperform several other traditional network embedding approaches [10]. Semi-supervised VAEs that incorporate partial label information [13]–[15], and VAEs that support interpretability using disentangled representation [16], [17] are among the works showing that VAE can be easy extended for different problem settings. The learned VAE representations are often used in node classification or network completion tasks [18], [19], [20], [21]. For example, there are works that aim to learn multi-modal network embedding using VAE as it can capture the correlation between different data modalities [18]–[20]. To co-embed the nodes and the auxiliary attributes in the same representation space, NetVAE employs a shared encoder to learn a unified representation for a nodes and its attribute, and separate decoders to decode them separately [21]. Another work that utilizes VAE to learn embeddings for nodes and attributes is CAN. It uses a double-VAE structure to jointly learn node and attribute representations [2]. Both NetVAE and CAN, however, do not consider external attribute semantics to improve node and attribute representations.

We compare six selected network embedding works in Table I. VGAE is a baseline model that learn the embedding based solely on node adjacency matrix, whereas the others utilize attribute or label association in the learning process. While TriDNR and ANRL learn the node representations with the help of attributes, the attribute representations are not included in the learned embedding models. On the contrary, CAN, NetVAE, and our proposed ECAN model both node and attribute representations at the same time. Furthermore, ECAN is the only model that considers attribute association and attribute semantics from external knowledge sources.

TABLE I: Summary of Related Works

Model	A Rep.	Encoding of				
		N Adj	N-L Assoc.	N-A Assoc.	A Adj	Ext. A Semantics
TriDNR [8]	×	√	√	√	×	×
ANRL [9]	×	√	×	√	×	×
VGAE [10]	×	√	×	×	×	×
CAN [2]	√	√	×	√	×	×
NetVAE [21]	√	√	×	√	×	×
ECAN	√	√	×	√	√	√

N: Node. **A:** Attribute. **L:** Label.

III. Co-Embedding with External Knowledge

A. Framework Overview

To provide a common embedding solution approach, we propose a **Co-embedding Attributed Networks with External Knowledge (CANE)** framework. As a framework, CANE allows different methods to be used in each step to realise different co-embedding attributed network models including ECAN. As illustrated in Figure 1, the CANE framework takes an attributed network $\mathcal{G}$ as input where each node is linked to one or more attributes. From $\mathcal{G}$, we extract useful information that serves as input to the learning of co-embedding model. The $N_{\mathcal{V}} \times N_{\mathcal{V}}$ adjacency matrix $\mathbf{A}$, and the $N_{\mathcal{V}} \times N_{\mathcal{A}}$ node attribute matrix $\mathbf{X}$ are derived from $\mathcal{G}$ directly, while attribute association matrix $\mathbf{S}$ is extracted from some external text corpus or knowledge graph. Our proposed extraction of attribute associations from text corpora and knowledge graphs will be described in Section III-B. Here, we assume that the external knowledge sources should be relevant to the given attributed network. With the derived attribute-to-attribute associations, the framework constructs the input matrices and learns the network and attribute embeddings as outlined in Sections III-C and III-D.

B. Attribute-to-Attribute Associations

Attribute Definition. The definition of attribute is important in our proposed framework as it determines how we extract attribute-to-attribute associations. Instead of defining attribute at the type-level, where different types of nodes are treated separately, our framework adopts an instance-level attribute definition. In other words, a node attribute is a concatenation of attribute type and attribute label(s). Using citation network as an example, a paper node can have (author, "John Smith") as an attribute, (keyword, "search engine"), and (keyword, "data mining") as two other attributes.

Depending on the attribute type, we derive the attribute-to-attribute associations differently. In this paper, we categorize attributes into 4 main types: (a) numeric, (b) categorical, (c) set, and (d) multiset. The attribute-attribute associations of numeric attributes (e.g., age, salary, etc.) are directly derived from their values. Hence, we focus on non-numeric attribute types in the following discussion.

Categorical attribute is a singleton set, and set is a special case of multiset. Without loss of generality, we only consider multiset attribute, and a node with multiset attribute is mapped to node-attribute associations, one for each value of the multiset. For example, a paper node v_i with this sentence in the abstract, "Social networks have long tails.", will be represented as a set of associations between v_i and (abstract,

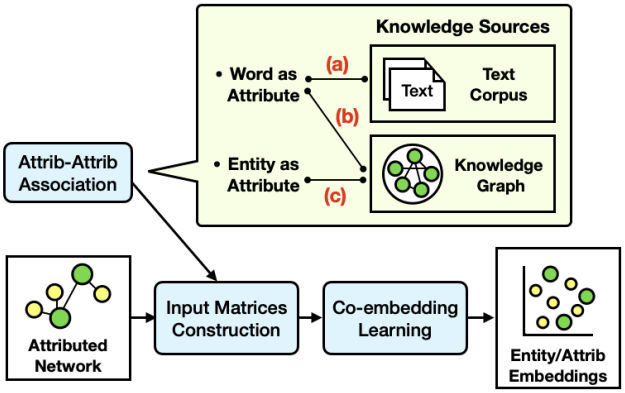

Fig. 1: Co-embedding Attributed Networks with External Knowledge

"social"), (abstract, "networks"), $\cdots$, (abstract, "tails"). In the following, we present two kinds of attribute values: (a) word, and (b) entity.

Word as Attribute. For attributes that are textual, words are the attribute values. As not all words are important, one can keep only the informative words and use them as attribute values. For example, we may select only words with high TF-IDF values.

Entity as Attribute. When the attribute has a category label or multiset of labels, each label is treated as an entity. Even when the attribute is textual, it is possible to extract entities from the textual content to be used as attribute values. Such an approach will be illustrated in our experiment datasets (see Section V-A).

With word and entity attributes defined, we now define measures for the association between two attributes. In this framework, we consider two types of external knowledge sources, (a) text corpus, and (b) knowledge graph.

(a) Text Corpus as External Knowledge. A text corpus consists of many text documents. The word-word associations can be derived from word co-occurrences, say, the number of time one word appears with another word in the same context. This is the same intuition behind popular word embedding approaches such as Skip-gram.

(b) Knowledge Graph as External Knowledge. Knowledge graphs are networks of entities connected by one or more semantic relation. From a knowledge graph, we could measure the association of two entities using the graph structure, the association between two words using their co-occurrences in knowledge graph's text data, and the association between a pair of entity and word using their co-occurrence. For instance, Milne and Witten proposed to measure the similarity of two entities by their in-link edges originating from same entities [22]. If two entities share very similar set of in-links, they are highly associated. Knowledge graphs, in many cases, have text data describing their entities. For example, Wikipedia has one article for each entity, and these entity articles can serve as a text corpus to derive entity-to-entity associations.

C. Input Matrices Construction

Next, we construct three set of input matrices for representing network and attribute-attribute associations before co-embedding learning.

1) Node-to-Node Association Matrix ($\mathbf{A}$)

The Node-to-Node Association Matrix captures the network structure of the attributed network using an $N_{\mathcal{V}} \times N_{\mathcal{V}}$ adjacency matrix. In the case of unweighted network, $\mathbf{A}_{ij} = 1$ if $(\mathcal{V}_i, \mathcal{V}_j) \in \mathcal{E}^{\mathcal{V}}$, and 0 otherwise. If the attributed network is weighted, $\mathbf{A}_{ij}$ will simply be the weight of edge.

2) Node-to-Attribute Association Matrix ($\mathbf{X}$)

The Node-to-Attribute Association Matrix is constructed for an unweighted attributed network by setting $\mathbf{X}_{ia} = 1$ if $(\mathcal{V}_i, \mathcal{A}_a) \in \mathcal{E}^{\mathcal{A}}$, and 0 otherwise.

3) Attribute-to-Attribute Association Matrix ($\mathbf{S}$)

There are three possible Attribute-to-Attribute Association Matrices, denoted by $\mathbf{S}^{(a)}$, $\mathbf{S}^{(b)}$, and $\mathbf{S}^{(c)}$ as shown in Figure 1.

(a) Word-to-word associations learned from text corpus ($\mathbf{S}^{(a)}$). In this setting, we extract contextual closeness between two attributes which are words from a text corpus $\mathcal{D}$. The text corpus could be relevant to the attributed network (e.g., the abstracts of the papers in a citation attributed network) or a text collection that provides meaningful word co-occurrence information. We derive the contextual closeness from distance between the two words a_i and a_j within a context window. For a window size w in a document d, the contextual closeness of one occurrence between a_i and a_j is $s^{\text{cx}}_{ijd} = w - \mathbf{d}_{ij}$ where $\mathbf{d}_{ij}$ denotes the distance between a_i and a_j. This closeness score is symmetric. For example, given a sentence "*Discriminative learning for differing training and test distributions*" and a window size 4, the contextual closeness between the words "learning" and "training" in this context window is $4-2=2$. The closeness between "learning" and "distribution" would be 0 as "distribution" does not appear within "learning"'s context window. For all pairs of words found in the attributed-network, we construct a $N_{\mathcal{A}} \times N_{\mathcal{A}}$ matrix $\mathbf{S}^{(a)}$ as $\mathbf{S}^{(a)}_{ij} = \sum_{d \in \mathcal{D}} s^{\text{cx}}_{ijd}$for all pairs of words a_i and a_j.

(b) Entity-to-entity associations learned from text corpus of knowledge graph ($\mathbf{S}^{(b)}$). Similar to (a), from the text corpus $\mathcal{D}$ that covers entities in a knowledge graph (e.g., Wikipedia pages), we construct a $N_{\mathcal{A}} \times N_{\mathcal{A}}$ matrix $\mathbf{S}^{(b)}$ using context closeness score s^{cx}_{ij} between two entities a_i and a_j. $\mathbf{S}^{(b)}_{ij} = \sum_{d \in \mathcal{D}} s^{\text{cx}}_{ijd}$

(c) Entity-to-entity associations learned from knowledge graph ($\mathbf{S}^{(c)}$). The corresponding $N_{\mathcal{A}} \times N_{\mathcal{A}}$ matrix $\mathbf{S}^{(c)}$ matrix has each entry measuring the relatedness between two entities. In this work, we define two entity relatedness functions giving rise to two different matrices:

(c.1) *In-link similarity from knowledge graph*: We adopt the Wikipedia Link-based Measurement (WLM) proposed in [22] to measure entity relatedness. Entities sharing similar in-link neighbors will have higher similarity, $\mathbf{S}^{(c1)}_{ij}$. WLM between two entities a_i and a_j is defined as follows, $\mathbf{S}^{(c1)}_{ij} = 1 -$

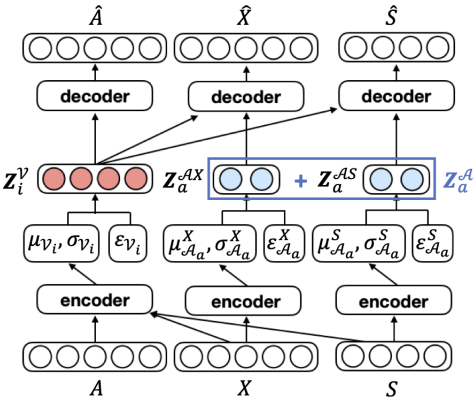

Fig. 2: ECAN (Non-Mixed Model)

$\frac{\log\max(|\mathcal{C}_{a_i}|,|\mathcal{C}_{a_j}|)-\log(|\mathcal{C}_{a_i}\cap\mathcal{C}_{a_j}|)}{\log|\mathcal{E}|-\log\min(|\mathcal{C}_{a_i}|,|\mathcal{C}_{a_j}|)}$ where $\mathcal{E}$ is the set of entities (attributes) in the attributed network, $\mathcal{C}_{a_i}$ is the set of entities having a link to a_i in the knowledge graph.

(c.2) *Cosine similarity from pre-trained knowledge graph embeddings*: This leverages on knowledge graph embedding methods such as Wikipedia2vec [23] to determine entity relatedness. Such embedding methods align entities and words in the knowledge graph in the same vector space. The association between two entities is this the relatedness between their vector representation. With a pre-trained knowledge graph embedding, we therefore define the association between entities a_i and a_j as $\mathbf{S}_{ij}^{(c2)} = \cos(\mathbf{a}_i^{\text{kg}}, \mathbf{a}_j^{\text{kg}})$. where $\mathbf{a}_i^{\text{kg}}$ denotes the vector representation of entity a_i in the pre-trained knowledge graph embeddings, and $\cos(\cdot)$ is the cosine similarity.

D. Co-Embedding Learning

The last step of our proposed framework is co-embedding learning. The co-embedding algorithm learns low-dimensional representation of nodes and attributes from the three aforementioned association matrices. After the embedding model is learned, we then use it in the subsequent downstream tasks. Our framework can accommodate any embedding learning algorithm that utilizes the same set of input matrices (i.e., node-node matrix, node-attribute matrix and attribute-attribute matrix) . In the remaining sections, we will use variational auto-encoders (VAE) as the embedding algorithm as illustrated in Section IV.

IV. Co-embedding Learning Using VAE

A. ECAN Models

Our proposed model **External Knowledge-Aware Co-Embedding Attributes Network (ECAN)** extends CAN with multiple VAEs used to encode and decode network edges $\mathbf{A}$, network attributes $\mathbf{X}$ and the new attribute semantics $\mathbf{S}$. This generalizes the two-VAE structure in CAN. As shown in Figure 2, besides a VAE for encoding/decoding network structure and network attributes, a new VAE is constructed to learn attribute information from attribute-to-attribute association matrix $\mathbf{S}$. The nodes are encoded by a two-layer GCN defined as:

$$\begin{aligned} H_{\mathcal{V}}^{(1)} &= \mathsf{ReLU}\left(\tilde{\mathbf{A}}\mathbf{X}\mathbf{S}\mathbf{W}_{\mathcal{V}}^{(0)}\right) \\ [\mu_{\mathcal{V}}, \sigma_{\mathcal{V}}^2] &= \tilde{\mathbf{A}}H_{\mathcal{V}}^{(1)}\mathbf{W}_{\mathcal{V}}^{(1)} \end{aligned} \tag{1}$$

where $\mathsf{RELU}(\cdot)$ is the layer-1 non-linear mapping from $\tilde{\mathbf{A}}\mathbf{X}\mathbf{S}$ to a hidden layer, and another layer of mapping to a set of distribution parameters $\mu_{\mathcal{V}}$ and $\sigma_{\mathcal{V}}$. $\tilde{\mathbf{A}}$ represents the normalized version of $\mathbf{A}$.

To encode the attributes, we encodes $\mathbf{X}$ and $\mathbf{S}$ separately in two VAEs, and concatenate the learned latent variables $\mu_{\mathcal{A}}$ and $\sigma_{\mathcal{A}}$ as attribute representation. The dimension size of these two VAEs are set to be $\frac{D}{2}$ so as to ensure the node and attribute representations could have the same length.

$$\begin{aligned} H_{\mathcal{A}}^{X(1)} &= \tanh\left(\mathbf{X}^T\mathbf{W}_{\mathcal{A}}^{X(0)} + b^{X(0)}\right) \\ H_{\mathcal{A}}^{S(1)} &= \tanh\left(\mathbf{S}\mathbf{W}_{\mathcal{A}}^{S(0)} + b^{S(0)}\right) \\ [\mu_{\mathcal{A}}^X, {\sigma_{\mathcal{A}}^X}^2] &= H_{\mathcal{A}}^{X(1)}\mathbf{W}_{\mathcal{A}}^{X(1)} + b^{X(1)} \\ [\mu_{\mathcal{A}}^S, {\sigma_{\mathcal{A}}^S}^2] &= H_{\mathcal{A}}^{S(1)}\mathbf{W}_{\mathcal{A}}^{S(1)} + b^{S(1)} \\ \mu_{\mathcal{A}} = [\mu_{\mathcal{A}}^X, \mu_{\mathcal{A}}^S],\ {\sigma_{\mathcal{A}}}^2 &= [{\sigma_{\mathcal{A}}^X}^2, {\sigma_{\mathcal{A}}^S}^2] \end{aligned} \tag{2}$$

In decoding, $\mathbf{A}_{ij}$ is generated from the product of $\mathbf{Z}_i^{\mathcal{V}}$ and $\mathbf{Z}_j^{\mathcal{V}}$. Attribute-wise, the node i's attribute a, $\mathbf{X}_{ia}$, is derived from the product of $\mathbf{Z}_i^{\mathcal{V}}$ and $\mathbf{Z}_a^{\mathcal{A}\mathbf{X}}$, and the association between attributes a and b, $\mathbf{S}_{ab}$, is derived from the product of $\mathbf{Z}_a^{\mathcal{A}\mathbf{S}}$ and $\mathbf{Z}_b^{\mathcal{A}\mathbf{S}}$. With three VAEs in our model, the loss function minimizes the reconstruction errors of $\mathbf{A}$, $\mathbf{X}$, and $\mathbf{S}$ as well as KL-divergence of the three approximated/true priors:

$$\begin{aligned} \log p(A, X, S) \geq\ & \mathbb{E}_{q_\phi} \sum_{i,j\in\mathcal{V}} \log_{p_\theta}(A_{ij} \mid Z_i^{\mathcal{V}}, Z_j^{\mathcal{V}}) \\ &+ \mathbb{E}_{q_\phi} \sum_{i\in\mathcal{V}, a\in\mathcal{A}} \log_{p_\theta}(X_{ia} \mid Z_i^{\mathcal{V}}, Z_a^{\mathcal{A}\mathbf{X}}) \\ &+ \mathbb{E}_{q_\phi} \sum_{a,b\in\mathcal{A}} \log_{p_\theta}(S_{ab} \mid Z_a^{\mathcal{A}\mathbf{S}}, Z_b^{\mathcal{A}\mathbf{S}}) \\ &- D_{KL}\left(q_\phi(Z^{\mathcal{V}}|A, X) \parallel p(Z^{\mathcal{V}})\right) \\ &- D_{KL}\left(q_\phi(Z^{\mathcal{A}\mathbf{X}}|X) \parallel p(Z^{\mathcal{A}\mathbf{X}})\right) \\ &- D_{KL}\left(q_\phi(Z^{\mathcal{A}\mathbf{C}}|S) \parallel p(Z^{\mathcal{A}\mathbf{S}})\right) \\ &\triangleq L(\theta, \phi; A, X, S) \end{aligned} \tag{3}$$

where

$$\begin{aligned} \mathbf{Z}_i^{\mathcal{V}} &= \mu_{\mathcal{V}_i} + \sigma_{\mathcal{V}_i}^2 \odot \epsilon, \text{ with } \epsilon \sim \mathcal{N}(0, I) \\ \mathbf{Z}_a^{\mathcal{A}\mathbf{X}} &= \mu_{\mathcal{A}_a}^X + {\sigma_{\mathcal{A}_a}^X}^2 \odot \epsilon, \text{ with } \epsilon \sim \mathcal{N}(0, I) \\ \mathbf{Z}_a^{\mathcal{A}\mathbf{S}} &= \mu_{\mathcal{A}_a}^S + {\sigma_{\mathcal{A}_a}^S}^2 \odot \epsilon, \text{ with } \epsilon \sim \mathcal{N}(0, I) \\ \mathbf{Z}_b^{\mathcal{A}\mathbf{S}} &= \mu_{\mathcal{A}_b}^S + {\sigma_{\mathcal{A}_b}^S}^2 \odot \epsilon, \text{ with } \epsilon \sim \mathcal{N}(0, I) \end{aligned} \tag{4}$$

The encoding process of ECAN makes use of both internal (network structure) and external (text corpuses/knowledge graph) knowledge. However, the correlation between $\mathbf{X}$ and $\mathbf{S}$ is neglected in such VAE structure. Therefore, we propose a mixed model inspired by AMVAE to cope with the joint encoding of different sources [20].

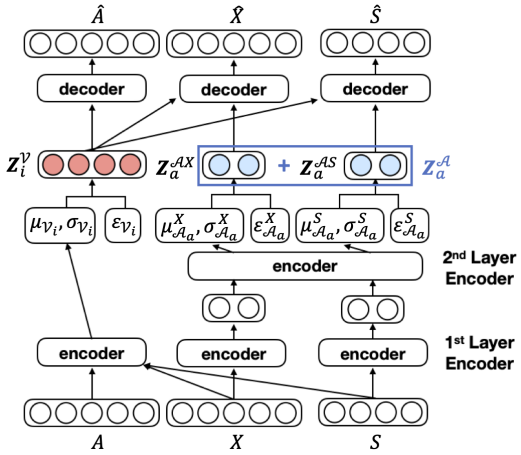

Fig. 3: ECAN (Mixed Model)

B. Mixed Model

When there are multiple inputs to the learning of representations, one intuitive approach is to concatenate the two input matrices as one input. Nevertheless, when one of the input matrices (say, S) has much larger dimension size than another (say, X), the encoding process might be dominated by the matrix with higher dimensions. To avoid this, the mixed model was proposed to jointly learn embedding from multimodal sources providing input data with different dimension sizes [20]. In the mixed mode, a two-encoder structure is deployed to balance the impact from different input.

We adopt the mixed model in our ECAN, and obtain a new ECAN structure called ECAN(mixed) (see Figure 3) which has an additional second-layer encoder that takes the concatenation of $\mathbf{X}$ and $\mathbf{S}$'s representations from the first-layer encoders as input. The second-layer encoder outputs the latent variables $[\mu^X_{\mathcal{A}_a}, {\sigma^X_{\mathcal{A}_a}}^2]$ and $[\mu^S_{\mathcal{A}_a}, {\sigma^S_{\mathcal{A}_a}}^2]$. Finally, we concatenate $\mathbf{z}_X$ and $\mathbf{z}_S$ derived from the latent variables to be the final attribute representation. The loss function of ECAN(mixed) is almost identical to ECAN except for the KL-divergence terms. Since the encoding of $\mathbf{Z}^{\mathcal{A}\mathbf{X}}$ and $\mathbf{Z}^{\mathcal{A}\mathbf{S}}$ is through a shared encoder, the approximated prior q_ϕ for the two latent variables should be derived from both $\mathbf{X}$ and $\mathbf{S}$. That is, the last two terms of Eq. 3 should be modified to $D_{KL}\left(q_\phi(Z^{\mathcal{A}\mathbf{X}}|X,S) \parallel p(Z^{\mathcal{A}\mathbf{X}})\right)$ and $D_{KL}\left(q_\phi(Z^{\mathcal{A}\mathbf{C}}|X,S) \parallel p(Z^{\mathcal{A}\mathbf{C}})\right)$ respectively. In this way, the correlation among input matrices is kept, and the learning is no longer dominated by the larger matrices.

V. Experiments

We conduct experiments on three datasets to evaluate the performance of our proposed ECAN model against CAN. In the earlier paper [2], CAN has been shown to outperform other state-of-the-art network embedding models including AANE [5], ANRL [24] and GAE [10] in both link prediction and node classification tasks on seven real datasets. In this section, we thus focus on evaluating ECAN against CAN in similar tasks. Our goal is to determine: (a) if external knowledge about attribute semantics can improve the embedding for achieving more accurate downstream task results (i.e., node classification and link prediction); (b) if the mixed model can yield better performance than non-mixed model; and (c) if there are evidence that better attribute and node embeddings are learned with the help of external knowledge sources.

TABLE II: Dataset Statistics

Datasets	#Nodes	#Edges	#Attrib (Words)	#Attrib (Entities)	#Labels
Citation	2,010	7,465	4,319	1,003	6
O*NET	974	11,342	18,119	1,174	6
IMDB	7,313	21,939	50,323	3,127	10

A. Datasets

Our experiments require attributed network datasets with words and entities as attributes. Associated with these datasets are suitable external knowledge sources covering associations between entities and between words. To the best of our knowledge, such datasets are not available and we decided to construct three new datasets specially for this research. While our models can handle multiple attributes, we keep the experiments simple and somewhat comparable to results in previous works by considering only one type of attribute. The dataset statistics are shown in Table II. Note that the node labels are only used in node classification task.

- **Citation Network Dataset [25].** This is a subset of the citation network dataset from ArnetMiner which has papers as nodes, and title combined with abstract as an attribute type. Two papers are connected if one cites another. The papers are classified into six domains based on the publication venues: database, AI, hardware, system, theory, and programming languages. A text corpus is constructed as an external knowledge source by combining every paper's title and abstract together to form a text document.
- **O*NET Occupation Dataset.** The Occupational Information Network (O*NET) (www.onetonline.org) is a free online occupation network created by the US government. Every occupation in O*NET is a node in the attributed network. Each occupation node has word attributes derived from its fields, namely: description, tasks and skills. An edge exists between two occupations in O*NET if they are related. In O*NET, every occupation is assigned a career preference profile according to the Holland Occupational Codes, also known as RIASEC [26]. There are six different RIASEC labels: Realistic (R), Investigative (I), Artistic (A), Social (S), Enterprising (E), and Conventional (C). Each occupation's profile consists of two to four of the above labels. A text corpus is constructed by combining the description, task and skill requirements of every occupation into a document.
- **IMDB Movie Dataset.** We constructed an attributed network by extracting a subset of movies from Internet Movie Database (IMDB). In this network, each node represents a movie, and movies are linked to one another by "related movies". Attributes are extracted from the movie synopsis. Every movie node is assigned one or

more movie genre: adventure, action, comedy, animation, family, fantasy, drama, sci-fi, romance, and mystery.

For each dataset, we construct both word attributes and entity attributes using their associated text corpora as follows. We extract word attributes from the text document associated with each network node (i.e., paper/occupation/movie node). For Citation dataset, we keep only the top 20 words with highest TF-IDF of each node as its word attributes. For both O*NET and IMDB datasets, we keep the top 30 words by TF-IDF as the node's word attributes. We next use spaCy to recognize entities in the text document associated with a node and match them with Wikipedia entities using Wikipedia2Vec [23] followed by manual judgement. The recognized entities include computing concepts and algorithm names for Citation dataset, and celebrity names and movie titles for IMDB dataset. For O*NET dataset, we link the skill entities in the text data of each occupations with Wikipedia entities by simple string matching. To select more important entities as attributes, we remove entities that appear in the text data of ≤ 10 nodes as we empirically found the choice of importance criteria not affecting the performance much.

B. Baseline and Model Settings

We compare several variants of the ECAN model using different attribute associations and mixed/non-mixed encoder model components, as enumerated below.

- **CAN [2]**: This double-VAE model learns the latent representations of nodes and attributes using $\mathbf{A}$ and $\mathbf{X}$ without external knowledge. CAN has been shown to outperform other state-of-the-art network embedding models including AANE [5] and ANRL [24] in link prediction and node classification tasks. Therefore, we use it as the baseline model for comparison.
- **ECAN(a)**: ECAN using $\mathbf{A}$, $\mathbf{X}$, and $\mathbf{S}^{(a)}$.
- **ECAN(b)**: ECAN using $\mathbf{A}$, $\mathbf{X}$, and $\mathbf{S}^{(b)}$.
- **ECAN(c1)**: ECAN using $\mathbf{A}$, $\mathbf{X}$, and $\mathbf{S}^{(c1)}$.
- **ECAN(c2)**: ECAN using $\mathbf{A}$, $\mathbf{X}$, and $\mathbf{S}^{(c2)}$.
- **ECAN(b+c)**: ECAN using $\mathbf{A}$, $\mathbf{X}$, $\mathbf{S}^{(b)}$, and $\mathbf{S}^{(c2)}$.

We show the result using mixed model structure and non-mixed model structure for all ECANs. Our ECANs are implemented using Tensorflow [27]. Adam optimizer is used for optimization with learning rate = 0.01 [28]. For the optimization of VAEs, we use the same parameter settings as CAN to allow these model to be comparable. D is set to be 32 for all datasets and all methods while the dimension size for the hidden layers is 64.

VI. Results

A. Node Classification

In this task, we train logistic regression models to predict node labels using the node representation as input. We use OneVsRest classifier for Citation Network Dataset as each node belongs to one class. For O*NET and IMDB datasets, we train 6 and 10 binary classifiers respectively as each node could have more than one class label. We report Macro-F1 using 10-fold cross validation following the previous works [5], [9]. As shown in Table III, our ECAN models outperform CAN which does not consider external knowledge sources. Among the ECAN models using different inputs, it is generally the case that (b+c) $>$ (c2) $\geq$ (c1) $>$ (b) $>$ (a). The result matches the intuition that (1) more attribute association information results in better performance; and (2) more clearly defined association (i.e., entity relatedness from knowledge graph) outperforms simple methods (e.g., contextual closeness among words). Finally, mixed models mostly outperform non-mixed ones. In particular, ECAN (Non-Mixed Model) can achieve 3% to 7.8% better macro-F1 than CAN, while ECAN (Mixed Model) can achieve 4% to 10% better macro-F1 than CAN. The improvement over the already quite accurate CAN highlights the importance of learning the mixed model that balances the correlation among different inputs.

TABLE III: Node Classification Result (Macro-F1)

	CAN	ECAN (Non-Mixed Model)				
		(a)	(b)	(c1)	(c2)	(b+c)
Citation	0.817	0.818	0.820	0.828	0.828	0.841
O*NET	0.801	0.813	0.832	0.839	0.841	0.851
IMDB	0.732	0.758	0.767	0.781	0.783	0.789
		ECAN (Mixed Model)				
		(a)	**(b)**	**(c1)**	**(c2)**	**(b+c)**
Citation	0.817	0.821	0.820	0.833	0.837	**0.849**
O*NET	0.801	0.819	0.832	0.841	0.850	**0.867**
IMDB	0.732	0.766	0.770	0.785	0.785	**0.804**

TABLE IV: Link Prediction Result (AUC)

	CAN	ECAN (Non-Mixed Model)				
		(a)	(b)	(c1)	(c2)	(b+c)
Citation	0.941	0.941	0.945	0.951	0.955	0.960
O*NET	0.897	0.903	0.911	0.915	0.916	0.923
IMDB	0.903	0.905	0.913	0.917	0.921	0.924
		ECAN (Mixed Model)				
		(a)	**(b)**	**(c1)**	**(c2)**	**(b+c)**
Citation	0.941	0.941	0.947	0.953	0.961	**0.964**
O*NET	0.897	0.909	0.912	0.919	0.919	**0.931**
IMDB	0.903	0.905	0.918	0.925	0.930	**0.937**

B. Link Prediction

Link prediction seeks to predict whether an edge exists between two given nodes. For each dataset, we train a binary Logistic Regression model that takes the Hadamard product of the two nodes' embeddings as input. Following the same procedure of positive/negative instances sampling as in the previous papers [2], [10], [29], we divide the existing node-node edges (i.e., positive instances) into three subsets: training set (85%), validation set (5%), and testing set (10%). We then randomly sample same amount of non-existed edges as the negative instances. As shown in Table IV, we report the Area under ROC curve (AUC) which has been used in most previous works. The result is consistent with that of node classification. ECAN(c2) is the best model among ECAN models using one type of attribute association. ECAN(b+c), again, outperforms all other ECAN and CAN models. In particular, ECAN(b+c) mixed model outperforms CAN by 2.4% to 3.8%, which is non-trivial given the little room for improvement.

TABLE V: Node Classification Performance (Macro-F1) of Different Dimension Size

Dataset / #dim	16	32	64	128
Citation	0.794	0.849	0.872	0.873
O*NET	0.8	0.867	0.87	0.871
IMDB	0.752	0.804	0.839	0.851

C. Impact of Varying Dimension Size

The dimension size, D, is important in VAEs as it could heavily affect the performance of downstream tasks in some cases. Hence, we conduct an experiment to examine how sensitive ECAN is to D. Here, we focus only on node classification tasks with ECAN(b+c). As shown in Table V, among all three datasets, as D gets larger, the node classification performance improves. We also observe that the improvement quantum diminishes as the dimension size increases. However, the diminishing improvement quantum varies with datasets. For instance, the improvement quantum of Macro-F1 from $D = 32$ and $D = 64$ is significant for both Citation and IMDB, while the Macro-F1 only improves by 0.3% for O*NET. When D reaches 128, the performance cannot improve further for both Citation and O*NET datasets, but there is still a 1.2% increase in Macro-F1 for IMDB dataset. Hence, one has to choose a suitable D setting for each dataset. O*NET dataset consists of a small number of nodes, thus we could represent all nodes with relatively small size of D (i.e., $D = 32$) and achieve good performance. As Citation and IMDB datasets are large, they need larger D values. In particular, IMDB dataset has the largest sets of nodes, attributes, and node labels. Therefore, even when the dimension size is large (i.e.,$D = 128$), the performance is yet to converge. In other words, there is a room for ECAN(b+c) to achieve better node classification accuracy if D is set to be larger.

VII. Acknowledgement

This research is supported by the National Research Foundation, Singapore under its International Research Centres in Singapore Funding Initiative. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore.

VIII. Conclusion

In this paper, we propose a framework that incorporates external knowledge sources, word and entity associations from external knowledge sources, to improve the quality of attributed network embedding learning. Based on the framework, we propose different multi-VAE co-embedding models with different encoder structures to incorporate these different association semantics. The experiment results show promising results of ECAN and its variants. This work represents an early effort to incorporate external knowledge in attributed network embedding. More combinations of external knowledge sources and attribute types can be considered in the future work. One can explore other alternative word embedding and knowledge graph embedding models to extract the association among attributes. Semi-supervised co-embedding models that can generate attributes of a node with some given class labels can also be explored. Finally, we can also improve the interpretability of ECAN by incorporating attention.

References

[1] X. Huang, J. Li, and X. Hu, "Label informed attributed network embedding," in *WSDM*, 2017.
[2] Z. Meng, S. Liang, H. Bao, and X. Zhang, "Co-Embedding attributed networks," in *WSDM*, 2019.
[3] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *KDD*, 2014.
[4] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *KDD*, 2016.
[5] X. Huang, J. Li, and X. Hu, "Accelerated attributed network embedding," in *SDM*, 2017.
[6] S. Wang, C. Aggarwal, J. Tang, and H. Liu, "Attributed signed network embedding," in *CIKM*, 2017.
[7] C. Yang, Z. Liu, D. Zhao, M. Sun, and E. Chang, "Network representation learning with rich text information," in *IJCAI*, 2015.
[8] S. Pan, J. Wu, X. Zhu, C. Zhang, and Y. Wang, "Tri-party deep network representation," in *IJCAI*, 2016.
[9] Z. Zhang, H. Yang, J. Bu, S. Zhou, P. Yu, J. Zhang, M. Ester, and C. Wang, "Anrl: Attributed network representation learning via deep neural networks." in *IJCAI*, 2018.
[10] T. N. Kipf and M. Welling, "Variational graph auto-encoders," 2016.
[11] K. Gregor, I. Danihelka, A. Graves, D. J. Rezende, and D. Wierstra, "Draw: A recurrent neural network for image generation," *arXiv preprint arXiv:1502.04623*, 2015.
[12] X. Yan, J. Yang, K. Sohn, and H. Lee, "Attribute2image: Conditional image generation from visual attributes," in *ECCV*, 2016.
[13] D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling, "Semi-supervised learning with deep generative models," in *NeurIPS*, 2014.
[14] W. Xu, H. Sun, C. Deng, and Y. Tan, "Variational autoencoder for semi-supervised text classification," in *AAAI*, 2017.
[15] Z. Meng, S. Liang, J. Fang, and T. Xiao, "Semi-supervisedly co-embedding attributed networks," in *NeurIPS*, 2019.
[16] D. Bouchacourt, R. Tomioka, and S. Nowozin, "Multi-level variational autoencoder: Learning disentangled representations from grouped observations," in *AAAI*, 2018.
[17] B. Esmaeili, H. Wu, S. Jain, A. Bozkurt, N. Siddharth, B. Paige, D. H. Brooks, J. Dy, and J.-W. van de Meent, "Structured disentangled representations," in *PMLR*, vol. 89, 2019, pp. 2525–2534.
[18] H. Li, H. Wang, Z. Yang, and H. Liu, "Effective representing of information network by variational autoencoder," in *IJCAI*, 2017.
[19] Z. Jiang, Y. Zheng, H. Tan, B. Tang, and H. Zhou, "Variational deep embedding: An unsupervised and generative approach to clustering," *arXiv preprint arXiv:1611.05148*, 2016.
[20] F. Huang, X. Zhang, C. Li, Z. Li, Y. He, and Z. Zhao, "Multimodal network embedding via attention based multi-view variational autoencoder," in *ICMR*, 2018.
[21] D. Jin, B. Li, P. Jiao, D. He, and W. Zhang, "Network-specific variational auto-encoder for embedding in attribute networks," in *IJCAI*, 2019.
[22] I. H. Witten and D. N. Milne, "An effective, low-cost measure of semantic relatedness obtained from wikipedia links," in *AAAI Workshop on Wikipedia and Artificial Intelligence: an Evolving Synergy*, 2008.
[23] I. Yamada, H. Shindo, H. Takeda, and Y. Takefuji, "Joint learning of the embedding of words and entities for named entity disambiguation," in *CoNLL*, 2016.
[24] H. Gao and H. Huang, "Deep Attributed Network Embedding," in *IJCAI*, 2018, pp. 3364–3370.
[25] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su, "Arnetminer: extraction and mining of academic social networks," in *KDD*, 2008.
[26] J. L. Holland, "A theory of vocational choice." *Journal of counseling psychology*, vol. 6, no. 1, p. 35, 1959.
[27] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: A system for large-scale machine learning," in *OSDI*, 2016.
[28] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *ICLR*, 2015.
[29] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.