WEAKLY-SUPERVISED NETWORK ALIGNMENT WITH ADVERSARIAL LEARNING

Nguyen Thanh Toan[1], Phan Thanh Cong[2] and Quan Thanh Tho[1]

[1]Department of Computer Science and Engineering, Ho Chi Minh City University of Technology, HCMC, Vietnam

[2]School of Information Communication Technology, Griffith University, Queensland, Australia

E-mails: nguyentoanit@gmail.com, thanhcong.phan@griffith.edu.au, qttho@hcmut.edu.vn

ABSTRACT

Network alignment, the task of seeking the hidden underlying correspondence between nodes across networks, has become increasingly studied as an important task to multiple network analysis. A few of the many recent applications of network alignment include protein network alignment, social network reconciliation, and computer vision. However, traditional methods which are based on matrix factorization directly work on networks themselves rather than exploit their intrinsic structural consistency, and thus their performance is sensitive to structural variations of networks. Recently, many supervised approaches which leverage latent representation have been proposed. Although they can handle large-scale datasets, most of them rely on a large number of parallel anchor links which are unavailable or expensive to obtain for many domains. Therefore, in this paper, we propose the WENA Framework, a representation learning-based network alignment, in which we study how to design weakly-supervised methods to align large-scale networks with a limit of ground truth available. Empirical results show that, with only two anchor links, WENA significantly outperforms existing unsupervised aligners and even outperforms state-of-the-art supervised methods that use richer resources in terms of both noise robustness and accuracy.

KEYWORDS

Network embedding, Graph mining, Network alignment, Graph matching, Knowledge representation

1. INTRODUCTION

Networks are universal languages for describing complex data. The network data structure naturally captures relationships between entities from different fields, such as social networks analysis, economics, bioinformatics and pattern recognition. Effective analyses on these data benefit a great range of subsequent research works and applications, including link prediction, node classification, and community detection, which prove the importance of network data. However, these works only focus on modeling single networks whereas many real-world applications require interpreting data from multiple networks. Consequently, in recent years, network alignment, the task of identifying the correspondence of nodes across different networks, has been proposed as a new research direction that aims to automatically identify hidden anchor links between networks to be prerequisite for many interesting inter-network applications.

In addition to its huge benefits, with the nature of a NP-hard problem [1], network alignment is challenging, though. Most natural approaches compute the network alignment by adopting a matrix factorization. Although these approaches are quite straightforward, it has been proven to solve the problem effectively. One famous algorithm is IsoRank [2], inspired by PageRank[3], which constructs an eigenvalue problem for every pair of input networks with the assumption that the protein in a PPI network is compatible with the protein in another network if the source node's neighbors match the neighbors of the target node. NetAlign [1] models the alignment problem as NP-hard combinatorial optimization problem and approximates it by applying the belief propagation heuristic. BigAlign [4] then reformulates bipartite network alignment as a new optimization problem and proposes a gradient-descent-based algorithm to solve it effectively. Again, on an optimization-based perspective, FINAL [5] also develop an algorithm to find the optimal alignment result by preserving network consistency principle. Using an approximation of low-rank matrix factorization, REGAL [6] further accelerates the network alignment process.

Inspired by network embedding techniques ([7],[8],[9],[10],[11]), many alignment approaches has been proposed under the theme of embedding and mapping approaches to deal with large-scale networks. Those approaches aim to induce network alignment work by independently learning the embeddings in each network using single network structure, and then learning a mapping from one embedding space into the other based on anchor links. The first of such methods is PALE [12] which learns nodes embedding by maximizing the co-occurrence likelihood of nodes then applies linear or multilayer perceptron (MLP) as mapping function. IONE [13] then uses the same mapping function as PALE but its embedding process is more complicated as it takes into account second-order node similarity. DeepLink [14] employs unbiased random walk to generate embeddings using skip-gram then using auto-encoder and MLP to construct mapping function. Whereas these methods put in a solid performance in some large datasets, they rely only on labeled data where the cost of generating them for a new machine learning task is often an obstacle for applying learning methods. To bridge this gap, in our end-to-end setting, we propose a GAN-based approach to bridge the gap so that our model is able to learn a mapping between the source and target network in the situation when very limit resources are available.

In this paper, we introduce a framework tilted **We**akly-supervised **N**etwork **A**lignment with adversarial learning (**WENA**). This framework is able to address network alignment problems when very limit resources are available. First, low-dimensional latent spaces are constructed independently based on the learning representation of the source and target networks. The learned embeddings preserve the semantic correlations of nodes in networks, hence, we treat them as features and build our alignment functions on top of them. We produce a mapping function $\mathbf{W}$ that roughly reconciles the spaces in the next step by mapping one embedding space to the other by employing a generative adversarial network for training the mapping function without cross-network supervision. The model is further improved by employing a weakly-supervised learning step using a limited number of anchor links. Finally, by using learned mapping $\mathbf{W}$, we leverage a greedy heuristic to seek for all hidden node pairs between networks. The main contributions are summarized as follows:

- We propose **WENA**, which identifies node alignments by learning node embeddings for each network and then, obtaining hidden node pairs by matching the embedding spaces.
- We propose a weakly-supervised network alignment approach that significantly outperforms other unsupervised methods. At the level of 10% network structural noise, our framework reaches over 90% whereas the best-unsupervised approach is just more than 50%.
- The performance of our method reaches or outperforms state-of-the-art supervised approaches although our framework works under very limited resource conditions.
- We conduct extensive experiments on real network datasets. Our framework not only achieves competitive performance but also exhibits robustness with network structural inconsistency.

The remaining sections of our paper are organized into six sections. Section 2 provides an overview of our approach. Section 3 outlines how to embed single networks independently, while Section 4 shows how to reconcile embedding spaces in the network alignment stage. Section 5 reveals the experimental results, and then we conclude the paper in Section 6.

2. Model and Approach

2.1. Model

In this section, we formally define the problem of network alignment in general. For simplicity, we focus on aligning two graphs (e.g., social or biological networks), although our method can easily be extended to more network types. Without loss of generality, we select one network as the source network and the other as the target network, denoted by $\mathbf{G}_s$ and $\mathbf{G}_t$ respectively. For each node in the source network, we aim to recognize, if any, its counterpart in the target network. To achieve this goal, network alignment techniques often calculate an alignment matrix $\mathbf{S}$, which is technically a cross-network similarity matrix: $\mathbf{S}(u, v)$ represents the similarity between a node $u \in \mathbf{V}_s$ and $v \in \mathbf{V}_t$. This can be formulated as follows:

Network alignment: *Given two networks* $\mathbf{G}_s = (\mathbf{V}_s, \mathbf{E}_s), \mathbf{G}_t = (\mathbf{V}_t, \mathbf{E}_t)$ *where* $\mathbf{V}_s, \mathbf{V}_t$ *are sets of nodes and* $\mathbf{E}_s, \mathbf{E}_t$ *are sets of edges, the problem of network alignment is to return an alignment matrix* $\mathbf{S}$ *where* $\mathbf{S}(u, v)$ *represents the similarity between a node* $u \in \mathbf{V}_s$ *and* $v \in \mathbf{V}_t$*.*

2.2. Approach

A common practical application of network alignment is to induce nodes from multi-networks that indicate to the same entity. Operationalizing this idea is challenging, though. Real-world networks often have a significant amount of nodes, therefore constructing the potential matching between two networks can be costly. In many cases, we fail to exploit supervised methods to achieve comparative performance as no or very limit anchor links exist between networks whereas other methods are insufficient or may mislead to poor alignment results. In our work, therefore, we study an alignment approach that leverages weakly-supervised representation learning to deal with network alignment which leads to both competitive accuracy and scalability.

Our framework overview is illustrated in Figure 1. Firstly, given the source and target networks as inputs, we encode them into a lower-dimensional space to reduce computational costs to subsequent tasks in the framework. The embedding step is crucial as this approach aims to handle large-scale networks. After the first stage, the networks are independently encoded so that good features are retained and facilitate the network alignment process, however, the distributions of these features may belong to different vectorial spaces. Hence, in the second step, we utilize the adversarial learning to minimize the KL divergence (relative entropy) between the two distributions (i.e, the source and the target embeddings) before forcing them much closer by a simple

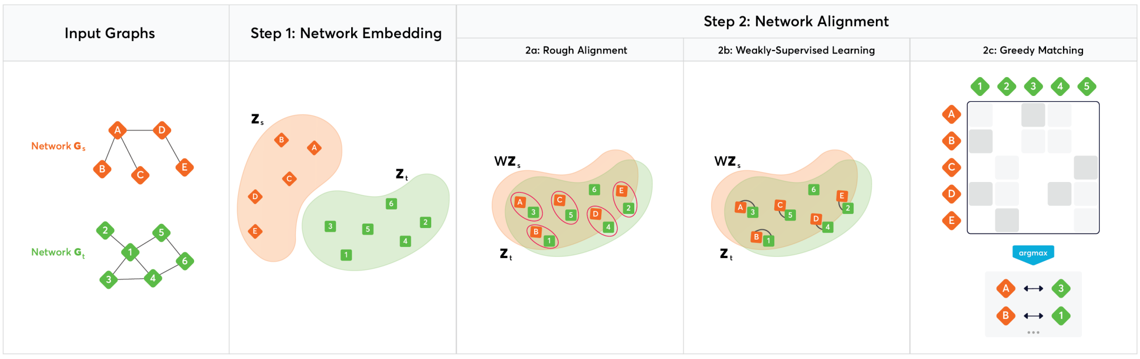

Input Graphs
Step 1: Network Embedding
Step 2: Network Alignment
2a: Rough Alignment
2b: Weakly-Supervised Learning
2c: Greedy Matching
Network G_s
Network G_t
argmax

Figure 1. **Overview of the WENA Framework**: Our framework first **(1)** conducts network embedding on each network to capture its major structural consistency, then based on adversarial training, the framework roughly aligns the node embeddings for source and target networks under an unsupervised setting **(2a)**. Afterward, the model can be further refined by seamlessly integrating with a weakly-supervised method **(2b)**, where only a limited number of aligned nodes are utilized as guidance. Finally, by using the learned mapping **W**, we leverage greedy matching **(2c)** to seek for all hidden node pairs between the networks.

matching function. This matching function is learned under weakly-supervised setting by employing two anchor links as guidance. In the final step, we apply a greedy heuristic to fast align all node pairs in the networks. The proposed framework consists of the following two main steps:

2.2.1. Network embedding

In this step, low-dimensional latent spaces are created independently based on learning representation of the source and target networks. We want to learn latent representations for network nodes so that more information is used to identify and distinguish the nodes across the networks. To efficiently learn the representations, we leverage a language-modeling technique. More detail of this step is described in Section 3.

2.2.2. Network alignment

Although the source and target networks are encoded by the same embedding technique, these embeddings might belong to different and incomparable vector spaces as the two encoding processes are independent. So the main task of network alignment is to learn a mapping function that reconciles the spaces by mapping one embedding space to the other. We first adopt an adversarial criterion to produce the mapping function that roughly approximates the source with target embedding. This mapping function is then further optimized by a simple weakly-supervised method to ensure that all embeddings are mapped into the common space. Finally, we leverage a greedy heuristic to align all nodes between graphs so that the embeddings from source and target networks that have similar representation will be chosen to form the aligned nodes. Greater detail of this step is described in Section 4.

3. NETWORK EMBEDDING

In this stage, we embed independently the two networks into a low-dimensional space. For simplicity, in this stage, we focus on embedding one network $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ in general without distinguishing between the source and the target network. For a pair of nodes $(u, v) \in \mathbf{E}$, with their embeddings (z_u, z_v), the probability to have an edge between them is:

$$p(u, v) = \sigma(z_u^T.z_v) = \frac{1}{1 + e^{-z_u^T.z_v}} \tag{1}$$

To learn the latent representation, we maximize the log-likelihood of all pair nodes in network $\mathbf{G}$:

$$\sum_{(u,v)\in\mathbf{E}} \log[\mathbf{P}(u, v)] = \sum_{(z_u,z_v)\in\mathbf{R^d}} \log[\sigma(z_u^T.z_v)] \tag{2}$$

To improve training efficiency, negative sampling is adopted. In particular, we maximize the objective function such as:

$$\log[\sigma(z_u^T.z_v)] + \sum_{k=1}^{K} \mathbf{E}_{v_k \propto \mathbf{P_n}(v)} \log[(1 - \sigma(z_u^T.z_k))] \tag{3}$$

where there are K negative samples. Empirically, each node is sampled with the probability $\mathbf{P}(v) \sim d_v^{3/4}$ as proposed in [15] and d_v is the degree of node v. Finally, we train the representation vector independently for all nodes of network $\mathbf{G}$ with stochastic gradient descent algorithm.

4. NETWORK ALIGNMENT

In this section, we consider that we have two embedded sets $(\mathbf{Z}_s \leftarrow \mathbf{G}_s, \mathbf{Z}_t \leftarrow \mathbf{G}_t)$ trained independently according to the method proposed in Section 3. Let $\mathbf{Z}_s = \{z_s^1, ..., z_s^n\}$ and $\mathbf{Z}_t = \{z_t^1, ..., z_t^m\}$ be two sets of n and m node embeddings which come from the source and target networks respectively. We focus on finding a mapping $\mathbf{W}$ between two sets so that $translations$

are close to each other in the shared space. We use the term $translation$ here as this technique is motivated by Conneau et al. [16] as an idea of machine translation. We propose a GAN-based approach to learn this mapping $\mathbf{W}$ between the source and target networks. We accomplish this by using both generative and discriminative processes.

4.1. Rough alignment under a GAN-based setting

The basic idea of a generative adversarial network (GAN) is a game between two players: the generator and the discriminator. Given a data distribution which we are trying to model p_{data}, the main goal of the first player, the generator, is to generate samples in p_{model} which approximates p_{data}. At the same time, the other player, the discriminator, examines samples to determine whether they are valid or fake data. The discriminator is trained as a traditional binary classifier under a supervised setting by dividing inputs into two classes: *real class* (label=0) or *fake class* (label=1). The two players in the game are exposed by two processes which are differentiable by their inputs and parameters:

- The discriminator is presented as a function that takes samples (both real and fake samples) as inputs and $\mathbf{\Theta}_D$ as parameters.
- The generator is a function that takes z_s as an input and uses $\mathbf{W}$ as a parameter. It is worth noting that the learned $\mathbf{W}$ of the second player will be used as the mapping function for network alignment. Hence, from now on, we call this process an aligner instead of a generator as described in the original GAN concept.

Both processes have cost functions defined on $\mathbf{\Theta}_D$ and $\mathbf{W}$:

- The cost function for the discriminator is $\mathbf{J}_D(\mathbf{\Theta}_D, \mathbf{W})$. By minimizing the $\mathbf{J}_D$, we train $\mathbf{\Theta}_D$ to maximize the probability of assigning correct labels to both training examples and samples from the generator. This process controls only $\mathbf{\Theta}_D$ as parameters and is independent on $\mathbf{W}$. The discriminator loss function can be written as:

$$\mathbf{J}_D(\mathbf{\Theta}_D|\mathbf{W}) = -\frac{1}{n}\sum_{i=1}^{n}\log \mathbf{P}_{\Theta_D}(\text{label} = 1|\mathbf{W}z_s^i) - \frac{1}{m}\sum_{j=1}^{m}\log \mathbf{P}_{\Theta_D}(\text{label} = 0|z_t^j) \quad (4)$$

where $z_s^i \in \mathbf{Z}_s, z_t^j \in \mathbf{Z}_t$.

- The cost function for the aligner is $\mathbf{J}_W(\mathbf{\Theta}_D, \mathbf{W})$. This process controls only $\mathbf{W}$ as parameters and is independent on $\mathbf{\Theta}_D$. In the unsupervised setting, $\mathbf{W}$ is trained so that the discriminator is unable to distinguish between generated and actual data. The aligner loss function can be written as:

$$\mathbf{J}_W(\mathbf{W}|\mathbf{\Theta}_D) = -\frac{1}{n}\sum_{i=1}^{n}\log \mathbf{P}_{\Theta_D}(\text{label} = 0|\mathbf{W}z_s^i) - \frac{1}{m}\sum_{j=1}^{m}\log \mathbf{P}_{\Theta_D}(\text{label} = 1|z_t^j) \quad (5)$$

where $z_s^i \in \mathbf{Z}_s, z_t^j \in \mathbf{Z}_t$.

4.1.1. Orthogonality

The aligner $\mathbf{W}$ obtained in the previous step gives a good performance. However, Xing et al. [27] showed that we can further improve the result by enforcing $\mathbf{W}$ orthogonal. This case is also instructed by Smith et al. [28]. They showed that the optimal linear transformation between vector spaces should be orthogonal as orthogonal matrices help maintain the distance between any two vectors across spaces. In our case, we also want to preserve the dot product of embeddings which forms a more stable training algorithm.

Although SVD guarantees us to achieve an optimal result by enforcing the constraint of orthogonality, this is an expensive procedure when we compute iteratively together with stochastic gradient descent. Furthermore, it may guide the parameters to go far from the optimizing direction of the main gradient descent. Hence, we use an approximate approach to make the algorithm more efficient (Cisse et al. [17]). In particular, after every main update, we perform the following update:

$$\mathbf{W} \leftarrow (1+\beta)\mathbf{W} - \beta(\mathbf{W}\mathbf{W}^T)\mathbf{W} \quad (6)$$

Empirically, we found that beta = 0.01 works best with our framework.

4.1.2. The training process

We follow the standard training procedure of Goodfellow et al. [18] to train our model. The training process consists of simultaneous mini-batch gradient descent. On each step, two mini-batches are sampled: a mini-batch of values from the target embedding and a mini-batch of values drawn from the source embedding. Then two gradient steps are made simultaneously: one updating $\mathbf{\Theta}_D$

to reduce $\mathbf{J}_D$ and one updating $\mathbf{W}$ to reduce $\mathbf{J}_W$. For each iteration, $\mathbf{W}$ is also updated according to orthogonal constraints after the main gradient update.

4.2. Weakly-supervised Learning

The above section introduces an unsupervised approach using an adversarial generative network to roughly align the embeddings. GAN has helped two data distributions to best match each other, however, it is inadequate to ensure the feasibility of the network alignment problem. We need to further guide the model with some cross-network anchor points so that the new aligner can be rotated to the right position that best supports network alignment. Specifically, we leverage a few labeled data (e.i., anchor pairs) as guidance, which yields a weakly-supervised approach. For the purpose of weakly-supervised learning, we apply the Procrustes solution on these known anchor links of pairs of embedded nodes and learn a linear mapping for source and destination space such that:

$$\mathbf{W}^* = \arg\min_{\mathbf{W}} ||\mathbf{W}\mathbf{A}_s - \mathbf{A}_t||_F \tag{7}$$

where $(\mathbf{A}_s, \mathbf{A}_t)$ are subsets of $(\mathbf{Z}_s, \mathbf{Z}_t)$ where anchor links exist. In that case, $\mathbf{W}$ in Equation 7 can be obtained by using singular value decomposition of $\mathbf{A}_t\mathbf{A}_s^T$:

$$\mathbf{W}^* = \arg\min_{\mathbf{W}} ||\mathbf{W}\mathbf{A}_s - \mathbf{A}_t||_F = \mathbf{U}\mathbf{V}^T \tag{8}$$

where $\mathbf{U}\Sigma\mathbf{V}^T = \mathbf{SVD}(\mathbf{A}_t\mathbf{A}_s^T)$. Empirically, we employ only two anchor links to guide the model and limit resources sufficient for the model to achieve competitive performance.

4.3. Greedy matching

The aligner $\mathbf{W}$ trained in the previous section is used to infer the similarity matrix as:

$$\mathbf{S} = (\mathbf{W}\mathbf{Z}_s)\mathbf{Z}_t^T \tag{9}$$

After getting the similarity matrix $\mathbf{S}$, a heuristic greedy matching algorithm [19] is applied on the matrix as a post-processing step to achieve one-to-one alignment accuracy. The heuristic iteratively finds the highest score $\mathbf{S}[i, j]$ on the similarity matrix and records the node pair (i, j), then all scores involving either node i or node j are deleted from the matrix (and replaced with a zero value). The process stops when one of the graphs has all of its nodes paired.

5. Experiments and Evaluations

In this section, we conduct experiments and evaluate the performance of our proposed **WENA** Framework.

5.1. Experimental setup

5.1.1. Datasets

To evaluate the efficiency and validity of our proposed framework, three real-world networks are adopted for experimental purposes, including protein-protein interaction (PPI) [11], economic network (ECON) [20], and brain network (BN) [21]. Network alignment methods primarily exploit the topology consistency of the network structure to perform alignment as nodes which share the same neighbors often tend to refer to the same entity. Following state-of-the-art alignment algorithms [5] [4], we study the structural noise by randomly removing edges from real-world datasets. Specifically, for each real network data set with an $\mathbf{A}_s$ adjacency matrix, we build a new network with the $\mathbf{A}_t = \mathbf{P}\mathbf{A}_s\mathbf{P}^T$ adjacency matrix, where $\mathbf{P}$ is a randomly generated permutation matrix with non-zero values that indicate the ground-truths for the network alignment problem. We add structural noise to $\mathbf{A}_t$ by removing edges with probability ranging from 0 to 0.2 without disassociating any nodes.

5.1.2. Baseline methods

We compare five well-known existing network alignment methods as summarized in Table 1.

Table 1. List of baseline methods

Technique	Abbreviation	Method
REGAL	REGAL	Unsupervised
FINAL	FINAL	Unsupervised
BigAlign	BigAlign	Unsupervised
PALE	PALE	Supervised
DeepLink	DeepLink	Supervised

5.1.3. Metrics

Recent approaches consider network alignment as a binary classification task. We simply employ a setwise metric [22] to define accuracy metric which is calculated by:

$$acc = \frac{\#\{\text{correctly identified node pairs}\}}{\#\{\text{ground truth node pairs}\}} \tag{10}$$

5.1.4. Settings

Due to the randomness, we run each data set 50 times to compute average results. For our discriminator, we use MLP with two hidden layers of size 2048, and Leaky-ReLU activation functions. The input to the discriminator is corrupted with dropout noise with a rate of 0.1. As suggested by Goodfellow [23], we include a smoothing coefficient s = 0.1 in the discriminator predictions. We use stochastic gradient descent with a batch size of 64, a learning rate of 0.1, and a decay of 0.98 both for the discriminator and $\mathbf{W}$. For other algorithms, we tune the parameters to have the best experiment performance.

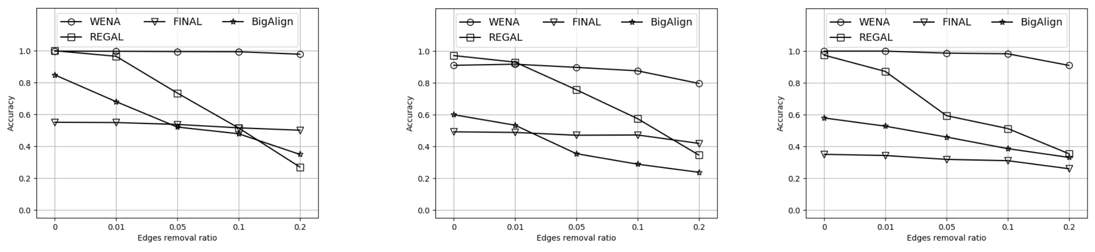
5.2. Alignment Performance Analysis

To assess the reliability of **WENA**, we empirically interpret the effectiveness of our weakly-supervised approach on several benchmarks and compare it with both state-of-the-art unsupervised and supervised methods. For each learning methods, we conduct a suitable analysis on both node correctness and robustness to structural noise.

5.2.1. Comparative performance to unsupervised methods

These experiments investigate comparative performance between different unsupervised models. Experimental results on the three datasets are presented in Figure 2.

5.2.1.1 Node correctness. The best existing unsupervised aligner we evaluate (REGAL) achieves an average accuracy of 32.28% for three datasets at the structural noise level of 0.2. With WENA we further improve it to 89.52%. We achieve an average improvement of 57.24% on node alignment when employing REGAL because while both algorithms use network structure information, REGAL adopts a strict assumption on topology consistency that the two nodes are similar when



(a) ppi dataset **(b) bn dataset** **(c) econ dataset**

Figure 2. Comparative performance of unsupervised methods across datasets

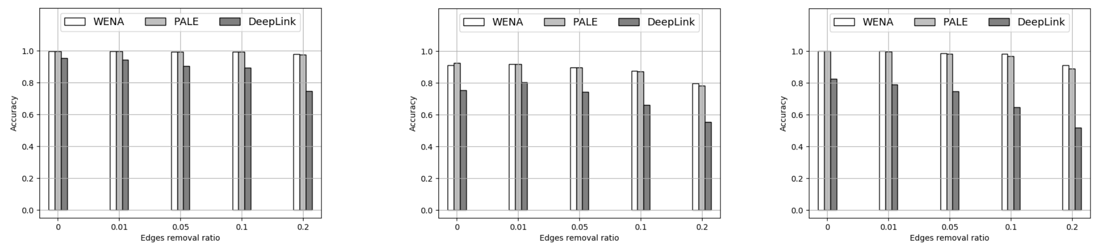
their neighbor's degree are the same, which makes its model susceptible to a considerable level of structure noise. The mean performance of FINAL and BigAlign on three datasets is 35.57% and 32.28%, respectively.

5.2.1.2 Robustness to structural noise. In general, all algorithms suffer accuracy drop when the noise level increased. WENA outperforms all existing methods. Furthermore, WENA is more robust to structural noise as the accuracy gap between it and other methods become larger when noise grows. Even when the noise level reaches 0.2, WENA is still able to achieve an average accuracy of over 90% for all three datasets. The performance of FINAL also witnesses a slight decrease but remains stable at an average level (with averages of 53%, 31% and 47% for PPI, ECON, and BN datasets, respectively). This is because FINAL depends heavily on priority matrices, while this information is rarely provided in real-world datasets and the default matrix generated by using nodes degree similarity is insufficient and affects the result. By contrast, the accuracy of REGAL and BigAlign drop sharply when the level of noise increases.

5.2.2. Comparative performance to supervised methods

These experiments investigate the comparative performance between different supervised models. The results are displayed in Figure 3, where the level of structural noise ranges from 0 to 0.2.

5.2.2.1 Node correctness. In this experiment, we use 20% of the number of known anchor links to guide supervised models (PALE, DeepLink). However, with very limited anchor links observation (as little as two anchor links), our proposed method is equal to or outperforms the existing supervised alignment methods. It can be seen from the results that WENA and PALE obtain the highest performance, while the accuracy of DeepLink is lower. Specifically, at 0.2 portions of noise, WENA is comparative with PALE and in most cases, their average accuracy is relatively



(a) ppi dataset **(b) bn dataset** **(c) econ dataset**

Figure 3. Comparative performance to supervised methods

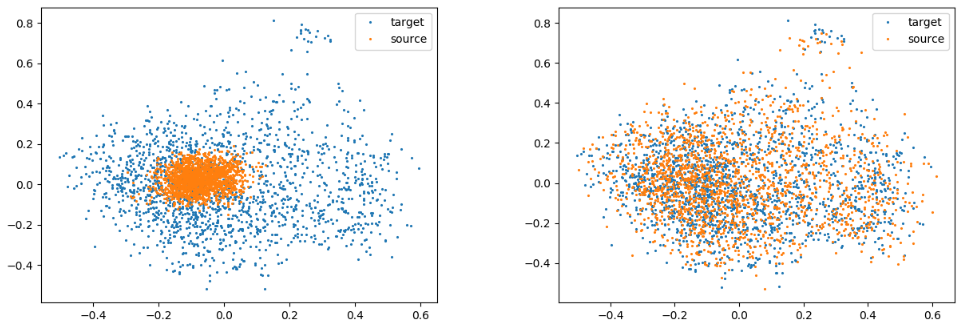
high (at nearly 90%). At the same level of noise (0.2), WENA achieves an average 28.8% improvement in terms of alignment accuracy over DeepLink.

5.2.2.2 Robustness to structural noise. It can be observed from the figures that WENA and PALE are stable when the level of noise increased. For instance, with the ECON dataset, WENA and PALE start with the same accuracy of 99% for noise-free cases and remain 92% and 90% respectively when the level of noise reaches 0.2. In contrast, DeepLink is sensitive to the structural noise factor. Its performance decreases gradually from 81% to 55% in the ECON dataset when noise increases from 0 to 0.2. The same phenomenon occurs with the BN dataset; its accuracy also decreased from 80% to 54% with the same range of noise observed. DeepLink seems to work better with the PPI dataset, however, the accuracy under 0.2 of noise is at 76%, which is still 22% lower when compared with WENA and PALE.

5.3. Case studies

In this section, we leverage the PCA algorithm to visualize the embedding distributions (i.e. project into 2D by taking the first two PCA components) at different stages: (before) - when no alignment method was performed and (after) - when our whole alignment process was incorporated into training. These visualizations should be viewed in color mode as the target embedding is in blue whereas the source embedding is in orange. We choose the BN dataset to intuitively validate the effect of the alignment algorithm.

Figure 4 shows that there is a strong correspondence between the two aligned data distributions. These intuitions contribute to the interpretation for the success of our algorithm both in terms of noise robustness and accuracy.



(a) Before alignment **(b) After alignment**

Figure 4. The effect of alignment step on embedding distribution (BN dataset).

6. Conclusions

In this paper, we propose a representation learning-based network alignment, called WENA. By leveraging adversarial training, our model roughly aligns nodes from a source network to those in a target network first under an unsupervised manner. Our approach can be further refined exploiting a few labeled data as guidance, leading to a weakly-supervised approach. The effectiveness of the proposed framework was evaluated on three real-work, large-scale networks. The reliability of WENA is verified by different benchmark datasets and different experimental settings.

Acknowledgement

This work was supported by the Domestic Master Programme of Vingroup Innovation Foundation (VINIF).

References

[1] M. Bayati, M. Gerritsen, D. F. Gleich, A. Saberi, and Y. Wang, “Algorithms for large, sparse network alignment problems,” in *2009 Ninth IEEE International Conference on Data Mining*, pp. 705–710, IEEE, 2009.

[2] R. Singh, J. Xu, and B. Berger, “Global alignment of multiple protein interaction networks with application to functional orthology detection,” *Proceedings of the National Academy of Sciences*, vol. 105, no. 35, pp. 12763–12768, 2008.

[3] W. Xing and A. Ghorbani, "Weighted pagerank algorithm," in *Proceedings. Second Annual Conference on Communication Networks and Services Research, 2004.*, pp. 305–314, IEEE, 2004.

[4] D. Koutra, H. Tong, and D. Lubensky, "Big-align: Fast bipartite graph alignment," in *ICDM*, pp. 389–398, 2013.

[5] S. Zhang and H. Tong, "Final: Fast attributed network alignment," in *KDD*, pp. 1345–1354, 2016.

[6] M. Heimann, H. Shen, T. Safavi, and D. Koutra, "Regal: Representation learning-based graph alignment," in *CIKM*, pp. 117–126, 2018.

[7] R. Perozzi, B.; Al-Rfou and Skiena, "Deepwalk: Online learning of social representations," in *KDD*, 2014.

[8] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *KDD*, pp. 855–864, ACM, 2016.

[9] X. Liu, N. Kertkeidkachorn, T. Murata, K.-S. Kim, J. Leblay, and S. Lynden, "Network embedding based on a quasi-local similarity measure," in *Pacific Rim International Conference on Artificial Intelligence*, pp. 429–440, Springer, 2018.

[10] H. Ji, C. Shi, and B. Wang, "Attention based meta path fusion for heterogeneous information network embedding," in *Pacific Rim International Conference on Artificial Intelligence*, pp. 348–360, Springer, 2018.

[11] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in Neural Information Processing Systems 30*, pp. 1024–1034, 2017.

[12] T. Man, H. Shen, S. Liu, X. Jin, and X. Cheng, "Predict anchor links across social networks via an embedding approach," in *IJCAI*, vol. 16, pp. 1823–1829, 2016.

[13] L. Liu, W. K. Cheung, X. Li, and L. Liao, "Aligning users across social networks using network embedding.," in *IJCAI*, pp. 1774–1780, 2016.

[14] F. Zhou, L. Liu, K. Zhang, G. Trajcevski, J. Wu, and T. Zhong, "Deeplink: A deep learning approach for user identity linkage," in *INFOCOM*, pp. 1313–1321, 2018.

[15] T. Mikolov, Q. V. Le, and I. Sutskever, "Exploiting similarities among languages for machine translation," *arXiv preprint arXiv:1309.4168*, 2013.

[16] A. Conneau, G. Lample, M. Ranzato, L. Denoyer, and H. Jégou, "Word translation without parallel data," *CoRR*, vol. abs/1710.04087, 2017.

[17] M. Cisse, P. Bojanowski, E. Grave, Y. Dauphin, and N. Usunier, "Parseval networks: Improving robustness to adversarial examples," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 854–863, JMLR. org, 2017.

[18] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems 27* (Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, eds.), pp. 2672–2680, Curran Associates, Inc., 2014.

[19] G. Kollias, S. Mohammadi, and A. Grama, "Network similarity decomposition (nsd): A fast and scalable approach to network alignment," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 12, pp. 2232–2243, 2012.

[20] R. A. Rossi and N. K. Ahmed, "The network data repository with interactive graph analytics and visualization," in *AAAI*, 2015.

[21] K. Amunts, C. Lepage, L. Borgeat, H. Mohlberg, T. Dickscheid, M.-É. Rousseau, S. Bludau, P.-L. Bazin, L. B. Lewis, A.-M. Oros-Peusquens, *et al.*, "Bigbrain: an ultrahigh-resolution 3d human brain model," *Science*, vol. 340, no. 6139, pp. 1472–1475, 2013.

[22] H. Zhang, M.-Y. Kan, Y. Liu, and S. Ma, "Online social network profile linkage," in *Asia Information Retrieval Symposium*, pp. 197–208, Springer, 2014.

[23] I. Goodfellow, "Nips 2016 tutorial: Generative adversarial networks," *arXiv preprint arXiv:1701.00160*, 2016.

AUTHOR BIOGRAPHY

Nguyen Thanh Toan is currently working as research assistant with Big Data and Smart Analytics Lab, Griffith University, Australia. His main research interests include the areas of database systems, decision support systems, and knowledge embedding.

Phan Thanh Cong is currently pursuing the master degree in Computer Science with Griffith University, Australia . His research interests include artificial intelligence, data mining, recommender systems, and big data analytics.

Quan Thanh Tho is an Associate Professor in the Faculty of Computer Science and Engineering, Ho Chi Minh City University of Technology, Vietnam. He received Ph.D degree in 2006 from Nanyang Technological University, Singapore. His current research interests include formal methods, program analysis, the Semantic Web, machine learning/data mining and intelligent systems.