Towards Higher-order Topological Consistency for Unsupervised Network Alignment

Qingqiang Sun\(^1\), Xuemin Lin\(^{2\dagger}\), Ying Zhang\(^3\), Wenjie Zhang\(^{1\dagger}\), Chaoqi Chen\(^4\)

\(^1\)University of New South Wales, Sydney, Australia \(^2\)Shanghai Jiao Tong University, Shanghai, China
\(^3\)University of Technology Sydney, Sydney, Australia \(^4\)University of Hong Kong, Hong Kong, China
qingqiang.sun@unsw.edu.au, xuemin.lin@gmail.com, ying.zhang@uts.edu.au, zhangw@cse.unsw.edu.au, chencq@connect.hku.hk

Abstract—Network alignment task, which aims to identify corresponding nodes in different networks, is of great significance for many subsequent applications. Without the need for labeled anchor links, unsupervised alignment methods have been attracting more and more attention. However, the topological consistency assumptions defined by existing methods are generally lower-order and less accurate because only the edge-indiscriminative topological pattern is considered, which is especially risky in an unsupervised setting. To reposition the focus of the alignment process from low-order to higher-order topological consistency, in this paper, we propose a fully unsupervised network alignment framework named HTC. The proposed higher-order topological consistency is formulated based on edge orbits, which is merged into the information aggregation process of a graph convolutional network so that the alignment consistencies are transformed into the similarity of node embeddings. Furthermore, the encoder is trained to be multi-orbit-aware and then is refined to identify more trusted anchor links. Node correspondence is comprehensively evaluated by integrating all different orders of consistency. In addition to sound theoretical analysis, the superiority of the proposed method is also empirically demonstrated through extensive experimental evaluation. On three pairs of real-world datasets and two pairs of synthetic datasets, our HTC consistently outperforms a wide variety of unsupervised and supervised methods with the least or comparable time consumption. It also exhibits robustness to structural noise as a result of our multi-orbit-aware training mechanism.

Index Terms—unsupervised network alignment, edge orbit, graph convolutional network, high-order topological consistency

I. INTRODUCTION

Network alignment task, which aims to identify entity correspondence across different networks, is usually the very first step of many downstream analyzing tasks. For instance, recognizing the same user on different social networks can facilitate friend suggestion, item recommendation, personalized advertisement [1]–[5]. Similar scenarios also exist widely in other fields, such as protein network analysis [6], knowledge discovery [7], etc.

Identifying corresponding nodes across different networks is an extremely hard task, even for humans. Manually labelling correspondence can be prohibitively challenging, expensive (in human efforts, time, and money costs), and tedious [8].

Due to such obstacles, in some cases, it may be impractical to get access to sufficient labels for training well-performed supervised or even semi-supervised models [4], [9]. By contrast, unsupervised models can be trained without the need for labeled data, which is more flexible and practical in real-world application scenarios. Thus, unsupervised alignment methods have been drawing a surge of interest recently [10]–[12].

In general, most existing alignment methods construct their models based on the assumption of attribute consistency or structural consistency, no matter supervised or unsupervised [13]. For instance, IsoRank [14] is a typical topology-only method that recognizes two nodes from two networks to be similar if their neighborhoods are similar. The most direct application of attribute consistency is to heuristically align users with the same user profiles such as name, gender, birthday, and location [15], [16]. Instead of relying on either attribute consistency or structural consistency, an increasing body of recent work makes efforts to take both attributes and structural information into consideration (if both are available) and receives significant performance gains by embracing the complementary effect between them [10], [17], [18].

Despite various types of topological information being utilized, such as node degree [18], connectivity [19], and co-occurrence in random walks [20], most existing works extract such information by relying solely on the most trivial topological pattern and accordingly define their low-order topological consistency. Specifically, in the trivial perspective of network topology, the connections between nodes are indiscriminative; that is, each edge plays the same role in the network. Nevertheless, edges can be further distinguished in a higher-order view of network topology based on their specific functions in local network structures, such as motifs [21] or graphlets [22] (it is worth noting that the so-called higher-order topology is not simply equivalent to a larger neighborhood). The main differences between trivial and high-order topology lie in two ways:

(a) Topological discrimination. Compared with trivial topology, high-order topology is more discriminative. For instance, as displayed in Fig. 1, all edges have the same weights/widths in the plain type of network topology, but they show a signifi-

}\(^{\dagger}\)Corresponding authors.
significant difference when higher-order edge patterns are considered (details about edge orbit will be illustrated in section IV). As a result, the local topology around a node can be distinguished to a greater extent.

(b) Accuracy in topological consistency. As the trivial topology is less discriminating, the corresponding topological consistency defined by that may not be accurate enough, which hence degrades the alignment performance. For example, say we want to identify the anchor node of node A in Fig. 1 based on their one-hop neighborhood. Any node with the same degree as node A can be regarded as satisfying low-order topological consistency according to the trivial topological pattern, which may be easily satisfied by a lot of candidates and thus lead to an ambiguous alignment. However, n the higher-order topology, a potential anchor node should not only have an identical degree but also have edges with similar weights connecting its neighbors, which results in a more accurate consistency and helps to filter out deceptive candidates.

Focusing solely on the trivial topology and corresponding consistency is generally more risky in an unsupervised setting. The fact is, models lack the capacity to extract extra information favourable to alignment accuracy from the trivial topology without the guidance of label information, and hence heavily depend on the consistency assumption. In other words, developing low-order topological consistency to high order is more crucial in this scenario.

To alleviate the aforementioned issues, in this paper, we propose a fully unsupervised framework called HTC (towards Higher-order Topological Consistency for unsupervised network alignment). Our assumption is that two nodes are more likely to have an anchor link if they hold not only low-order but also higher-order topological consistency. Considering that, we define higher-order topological consistency based on edge orbits. By conducting an orbit-weighted aggregation process, we transform the higher-order topological consistency and attribute consistency into the similarity of corresponding node embeddings, which is theoretically analyzed and proved. Without labeled data, we learn from the training paradigm of Graph Auto-Encoder (GAE) [23] but share the encoding parameters between different orders of topology so that the encoder can be multi-orbit-aware. In addition, we propose an effective fine-tuning mechanism to find more trustworthy anchor links by assigning trusted node pairs a larger aggregation coefficient, and its validity is verified in theory. A comprehensive evaluation of node correspondence is obtained by further integrating alignment scores computed based on different orders of consistency, rather than focusing on any single topological pattern.

The proposed HTC framework is not only theoretically sound but also empirically performs well. HTC consistently outperforms a wide variety of state-of-the-art models, even though some of those competitors are supervised, across three pairs of real-world datasets with the least or comparable time cost. In particular, significant improvements (up to 49% in terms of precision@1) over the basic Graph Convolutional Network (GCN) based alignment paradigm are observed continuously. Experimental results on two pairs of synthetic datasets also verify the robustness of HTC against structural noise. Other experimental results, including ablation tests, hyperparameter studies, and visualization analyses, further verify the significance of the proposed higher-order topological consistency in improving alignment precision.

Our contributions can be summarized as follows:

- We propose a fully unsupervised network alignment framework, which formulates higher-order topological consistency based on edge orbits.
- It is theoretically proved that the formulated higher-order topological consistency combined with attribute consistency can derive node embedding similarity, which transforms the network alignment task into the task of node embedding similarity measurement.
- As an additional effect of the proposed multi-orbit-aware training mechanism, our method can be robust to structural noise.
- We introduce the concept of trusted pairs and accordingly refine embeddings so as to find more trusted pairs, which can alleviate the hubness problem that accompanies the roughly-learned embeddings.
- The superiority of our method, such as effectiveness, efficiency, and robustness, is comprehensively evaluated through extensive experiments.

The rest of this paper is organized as follows. We briefly review related work in Section II. The problem is formulated in Section III before proposing our framework in Section IV. Section V reports experimental results, followed by conclusions made in Section VI.

II. RELATED WORK

A. Network Alignment

So far, most network alignment approaches are supervised. Supervisory data are required so as to refine embedding space either in the form of partial ground truth (e.g. PALE [19], CENALP [24]) or prior alignment matrix (e.g. IsoRank [14], FINAL [17]). Due to the expensive cost of manual labeling, unsupervised methods are more suitable and desirable in some real-life applications.
Another noticeable trend is that more and more methods try to leverage both structural and attribute information so as to better capture the information of the network nodes (such as FINAL [17] and REGAL [18]). Recently, some work has taken advantage of the natural strength of GCN in integrating both network topology and node attributes and has empirically shown the great potential of GCN for alignment tasks [10], [25]. A wide variety of structural metrics are proposed and utilized for formulating corresponding topological consistency, like node degree [18], connectivity [19], and co-occurrence in random walks [20]. Besides, some methods pay attention to a broader neighborhood of nodes, such as stacking more GCN layers [10] or aligning triangular structures [26]. Even so, most previous work essentially just models the edge-indiscriminative topological pattern, and hence their corresponding topological consistencies remain low-order. Unlike previous work, we formulate the high-order topological consistency based on different orders of edge pattern. It should be emphasized that larger neighborhoods and higher-order consistency cannot be simply equalized. The experimental results also empirically demonstrate that our proposed higher-order topological consistency performs far better than simply considering a larger neighborhood, such as introducing diffusion matrices [27].

B. High-order Pattern

High-order patterns of network structure such as graphlets and motifs have been demonstrated to be very informative and helpful for different tasks in many fields. For instance, graph motifs are utilized in [28] for classifying directed CORA citation networks. Graphlet patterns are taken into consideration to obtain better performance in the semi-supervised node classification task [29]. In [30], the use of motifs enables the perception of high-order semantics in heterogeneous graphs. In more detail, each node or edge can be differentiated at the level of orbit [31], [32]. Some network alignment techniques, such as H-GRAAL [33], GREAT [34], GraphletAlign [35], etc., treat the graph degree vectors as node/edge features and then quantify the corresponding topological similarity for network alignment. By contrast, we define high-order topological consistency based on the constructed graph orbit matrix, which is injected into the aggregation process of GCN in order to further extract and integrate useful information. In this way, we are capable of obtaining abstract features of nodes for similarity computation.

C. Graph Convolutional Network

Compared with other node embedding techniques like Node2Vec [36], DeepWalk [37], and LINE [38], GCN [39] is naturally capable of integrating both structure and attribute information due to its special encoding mechanism and has shown its effectiveness in processing graph-structured data [40]. To address the issue that GCN treats every edge equally without distinction, Graph Attention Network and its variants are proposed, which generally need labels to train extra parameters [29], [41]. However, by using higher-order topological patterns, we can distinguish edges without introducing additional training-required parameters, which is a great merit for unsupervised learning.

Auto-Encoder and its variants have important applications in many fields. For example, Wang et al. [42] use variational autoencoder with dynamic extensions to successfully address the dynamic interaction problem of latent variables. As a special variant, Graph Autoencoder (GAE) [23] is proposed for unsupervised learning on graph-structured data. Inspired by such an encoder-decoder paradigm, we train our GCN encoder to be multi-orbit-aware by sharing encoding parameters among different orders of orbit topology. Hence, it can be expected to prevent the model from relying on any single topological pattern.

III. PROBLEM FORMULATION

Our work targets the problem of aligning two attributed networks. Generally, an attributed network can be represented as \( G = (V, A, X) \), where \( V \) is a set consisting of \( n \) nodes, \( A \in \{0,1\}^{n \times n} \) is the adjacency matrix of the network which contains the connection details among those nodes, and \( X \in \mathbb{R}^{n \times d} \) is the attribute/feature matrix in which every node has a \( d \)-dimensional feature.

Typically, one of two to-be-aligned networks / graphs is called source network / graph and the other is called target network / graph, denoted as \( G_s = (V_s, A_s, X_s) \) and \( G_t = (V_t, A_t, X_t) \) respectively. The correspondence is referred to as anchor links, and the nodes forming the links are referred to as anchor nodes. Mathematically, the unsupervised network alignment task can be defined as follows:

**Definition 1. Unsupervised Network Alignment.**

Given a source network \( G_s = (V_s, A_s, X_s) \) and a target network \( G_t = (V_t, A_t, X_t) \), the objective of unsupervised network alignment is to identify all possible anchor links across \( G_s \) and \( G_t \) by computing the alignment matrix \( M \in \mathbb{R}^{n_s \times n_t} \) without using any observed anchor links, where every entry \( M(i,j) \) represents the alignment score between node \( i \in V_s \) and node \( j \in V_t \).

![Fig. 2: An illustrative example of alignment consistency. Attribute Consistency: the gender attribute of node “a” and its corresponding node “A” are both “male”; Topology Consistency: nodes “a” and “b” are connected in \( G_s \), and their corresponding nodes “A” and “B” are also connected in \( G_t \). Generally, most existing work follows the consistency restraints shown in Fig. 2.](image-url)

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
IV. METHODOLOGY

The framework of the proposed HTC is shown in Fig. 3. We introduce details of HTC in this section.

A. Higher-order Topological Consistency

We first construct graphlet orbit matrix and define the higher-order topological consistency based on it.

Graphlet is a term used to denote a connected network with a small number of nodes [22]. Any induced subgraph can be grouped into a certain type of graphlet. \( G' = (V', E') \) is an induced subgraph of \( G = (V, E) \) if and only if \( V' \subseteq V \) and \( E' = \{(u, v) \in E \mid u, v \in V'\} \). Furthermore, the edges of each graphlet can be clustered into orbits with respect to the graphlet automorphisms [32]. For instance, there are 9 graphlets and 13 edge orbits for subgraphs with 2-4 nodes as shown in Fig. 4. Orbits define the “roles” of edges within the graphlet. As an example, in the graphlet of a three-edge chain, one edge represents the bridge and the remaining two edges can be connected through the bridge; the edges of this graphlet thus form two different orbits (numbered 4 and 3, respectively). Moreover, edges around a node can be distinguished by comparing their occurring frequency on multiple orbits, as in the illustrative example shown in Fig. 5.

We construct a set of graphlet orbit matrix (GOM), denoted as \( \mathcal{O} = \{O_0, O_1, \ldots, O_K\} \), where \( K \) matrices correspond to \( K \) orbits. Each GOM contains the frequency of edges on the corresponding orbit. Formally, given a graph \( G \) with a vertex set of size \( n \) and any two vertices \( i \) and \( j \) of it, the \( k \)-th GOM \( O_k \in \mathbb{N}^{n \times n} \) is defined as follows:

\[
O_k(i, j) = \text{the number of times that edge } (i, j) \text{ occurs on orbit } k. \tag{1}
\]

The greater the value of \( O_k(i, j) \) is, the stronger the connection between node \( i \) and node \( j \) is with respect to the orbit \( k \). In addition to the above weighted definition, GOM can also be binary, in which \( O_k(i, j) \) equals to 1 as long as the edge \( (i, j) \) is observed on orbit \( k \) for at least one time, otherwise the value of \( O_k(i, j) \) is 0. In the binary case, the distinctions between edges are weakened. Referring to the example in Fig. 5, the value of edge \((b, c)\) on orbit 1 would be 1, which is the same as that of edge \((a, b)\) if using binary setup. Hence, in this paper, we mainly consider the weighted form of GOM.

Taking the value of each entry in GOM as the weight of the corresponding edge, GOMs exactly represent different levels of induced networks, as shown in Fig. 1. Therefore, we can mathematically define higher-order topological consistency based on GOMs:

**Definition 2.** High-order Topological Consistency.

Given two nodes \( u, v \in V \), and their corresponding nodes \( u', v' \in V' \), they are regarded as satisfying \( k \)-order topological consistency, if their edges have identical weights in terms of the \( k \)-th GOM, i.e., \( O_{k, k}(u, v) = O_{k, k}(u', v') \).

B. Orbit-weighted Encoding

To integrate attribute consistency and higher-order topological consistency, we elegantly merge higher-order topological information into the aggregation process of GCN. Instead
of treating all edges equally, we respect the fact that edges playing different roles in each high-order structure should be given different weights when passing messages.

**Weighted aggregation.** We naturally assign the orbit-defined weight to an edge for passing information between nodes. Formally, the feature of node $i$ on $t$th layer and $k$th orbit can be obtained by:

$$ h_{k,i}^t = f \left( \sum_{j \in N(i)} O_k(i,j) h_{k,j}^{t-1} W_{k}^{t-1} \right), $$

which can be performed efficiently in matrix formulation:

$$ H_k^t = f \left( O_k H_k^{t-1} W_{k}^{t-1} \right), $$

where $H_k \in \mathbb{R}^{n \times d}$ is a feature matrix, $f$ is a nonlinear activation function, and $W_{k}^{t-1}$ is a trainable weight matrix for node representation encoding.

**Modified self-connection.** In Eq. (2), only information from neighbors is considered to generate a node’s feature. To include the information from a node itself, we need to add self-connections. Note that the values of a weighted orbit matrix denote the frequency of edges occurring on a certain orbit, which may be far bigger than 1. In this case, using a typical identity matrix as the self-connection matrix is likely to result in the issue that the impact of self-connection is significantly weakened compared with its neighbors. Here, we define a more appropriate self-connection matrix with respect to orbit $k$ as follows:

$$ C_k(i,i) = \begin{cases} 1, & \text{if } \max_{1 \leq j \leq n} O_k(i,j) = 0; \\ \max_{1 \leq j \leq n} O_k(i,j), & \text{otherwise.} \end{cases} $$

where $\max_{1 \leq j \leq n} O_k(i,j) = 0$ means that node $i$ is not connected to any other node on orbit $k$. Intuitively, this matrix allows the node to either assign an importance to itself that is in line with that of its most important neighbor, or just consider its own information if there are no neighbors around it. Hence, the orbit matrix can be modified by $\tilde{O}_k = O_k + C_k$.

**Normalized Laplacian.** According to the aggregation mechanism of GCN, nodes with large values in $\tilde{O}_k$ will have large values in their feature representation, and vice versa. This may cause vanishing or exploding gradients [39]. Hence, in practice, the procedure of normalization is required. The symmetric normalized Laplacian matrix on orbit $k$ can be obtained by $\tilde{L}_k = \tilde{F}_k \tilde{F}_k^\top$, where $\tilde{F}_k(i,i) = \sum_{j=1}^{n} O_k(i,j)$ is a diagonal frequency matrix.

**Forward encoding.** Say the GCN encoder contains $L$ hidden layers, the output embeddings of two graphs, denoted as $H_{2,k}^L$ and $H_{1,k}^L$, respectively, can be obtained through the forward encoding process:

$$ H_{2,k}^L = f^{L-1} \left( \tilde{L}_{k,1} \cdots f^0 \left( \tilde{L}_{k,0} X_s W_s^0 \right) \cdots W_s^{L-1} \right) , $$

$$ H_{1,k}^L = f^{L-1} \left( \tilde{L}_{k,1} \cdots f^0 \left( \tilde{L}_{k,0} X_t W_t^0 \right) \cdots W_t^{L-1} \right) , $$

where the non-linear activation functions $F = \{f^0, f^1, \cdots, f^{L-1} \}$ and the weight parameters $W = \{W_s^0, W_t^1, \cdots, W_t^{L-1} \}$ are shared between source and target graphs.

**Theoretical Analysis.** We mathematically prove that high-order structural consistency and attribute consistency will lead to identical node embeddings across two networks through our proposed encoding mechanism. First, we have the following lemma.

**Lemma 1.** For two nodes in the same graph $u$ and $u'$, if there exists a matching set $M = \{(v, v') | v \in N(u), v' \in N(u') \}$ and nodes of each matching pair have identical attributes, i.e. $X(v) = X(v')$ and identical topological patterns in terms of orbit $k$, i.e. $O_k(u,v) = O_k(u',v')$, then the embeddings of $u$ and $u'$ are identical after encoded by one GCN layer, i.e. $H_k(u) = H_k(u')$.

**Proof.** The embeddings of node $u$ and $u'$ can be wrote as:

$$ H_k(u) = f^0 \left( \tilde{L}_k(u,:) X W^0 \right) $$

$$ H_k(u') = f^0 \left( \tilde{L}_k(u',:) X W^0 \right) $$

Since $O_k(u,v) = O_k(u',v')$, the embeddings of $u$ and $u'$ after encoded by the shared GCN parameters are identical, i.e. $H_k(u) = H_k(u').$ □

Actually, Lemma 1 indicates that orbit-defined aggregation inherits the characteristics of the typical aggregation manner of GCN that attribute and topology similarity of nodes can be transformed into the similarity of node embeddings. To leverage the power of orbit-weighted aggregation for alignment tasks, we extend Lemma 1 to the scenario of two networks and obtain the following proposition.

**Proposition 1.** Suppose that two nodes $u \in V_s, u' \in V_t$ have an anchor link between them, and there exists a matching set $M = \{(v, v') | v \in N_s(u), v' \in N_t(u') \}$. All matching pairs $(v, v')$ satisfy the attribute consistency, i.e. $X_s(v) = X_t(v')$ and the $k$-order topological consistency, i.e. $O_{s,k}(u,v) = O_{t,k}(u',v')$, then the embeddings of $u$ and $u'$ after encoded by the shared GCN parameters are identical, i.e. $H_{s,k}(u) = H_{t,k}(u')$.

**Proof.** Similar to the proof of Lemma 1, we first expand the expression of embeddings of node $u$ and $u'$:

$$ H_{s,k}(u) = f^0 \left( \tilde{L}_{s,k}(u,:) X_s W_s^0 \right) $$

$$ H_{t,k}(u') = f^0 \left( \tilde{L}_{t,k}(u',:) X_t W_t^0 \right) $$

According to attribute consistency and topological consistency in terms of orbit $k$, we have $X_s(v) = X_t(v')$ and $L_{s,k}(u,v) = L_{t,k}(u',v').$
\[ \hat{L}_{\tau,k}(u', v'). \] Besides, as the GCN encoder is shared between source and target graphs, \( f_0^s = f_0^t \) and \( \hat{W}_s^0 = \hat{W}_t^0 \) are satisfied. Hence, \( H_{\tau,k}(u) = H_{\tau,k}(u') \). Similarly, for \( L \)-layer GCN, \( H_{\tau,k}^L(u) = H_{\tau,k}^L(u') \) holds if corresponding \( L \)-hop neighborhood is consistent.

The key factor in extending Lemma 1 to Proposition 1 is the sharing of GCN encoder between two networks, which guarantees that nodes in two networks are embedded by the same mapping mechanism. Therefore, the consistency constraints of alignment can be transformed into the similarity of node embeddings, and an extra step for aligning feature space can be omitted, which is claimed to be necessary in some previous embedding-based methods [43].

C. Multi-orbit-aware Training

As opposed to just using the trivial edge-defined topology information, multiple higher-order topological patterns give us more comprehensive information for aligning nodes.

**Orbit-reconstruction loss.** Without alignment labels, we adopt auto-encoder scheme to train parameters \( \mathcal{W} \). We reconstruct each orbit Laplacian matrix using generated representations. The matrix of source graph on orbit \( k \) is rebuilt by:

\[
\hat{L}_{s,k} = H_{s,k}^L \cdot H_{s,k}^{-1}. \tag{6}
\]

The reconstructed Laplacian matrix of target graph \( \hat{L}_{t,k} \) can also be obtained in a similar way. The orbit-reconstruction loss for orbit \( k \) is further computed as follows:

\[
\text{loss}_k = \left\| \hat{L}_{s,k} - \tilde{L}_{s,k} \right\|_F + \left\| \hat{L}_{t,k} - \tilde{L}_{t,k} \right\|_F, \tag{7}
\]

where \( \left\| \cdot \right\|_F \) denotes the Frobenius norm which is used to measure the difference between the original Laplacian matrix \( \hat{L}_{s,k} \) and the reconstructed one \( \tilde{L}_{s,k} \). By minimizing \( \text{loss}_k \), the GCN encoder can be regarded as \( k \)-th-orbit-aware since the useful information required for reconstructing the \( k \)th orbit matrix is captured.

To be multi-orbit-aware, all orbit-reconstruction loss items are added up to the overall objective:

\[
\Gamma = \sum_k \text{loss}_k. \tag{8}
\]

As \( \Gamma \) is minimized by gradient descent optimizers, like Adam [44], the similarity of the embeddings generated by the encoder is able to reflect the corresponding topological and attribute consistency. The multi-orbit-aware process is summarized in Algorithm 1.

D. Trusted-pair based Fine-tuning

In this subsection, we use \( h_s \) and \( h_t \) to denote the embeddings of source graph and target graph output by the last layer on any orbit, omitting the superscripts \( L \) and subscripts \( k \) for brevity.

**Hubness problem.** By Lemma 1 and Proposition 1, both potential anchor nodes across different graphs and nodes with similar attributes and structure in the same graph will have embeddings as similar as possible. In high dimensional embedding space, these similar embeddings will be located closely. As a result, some nodes, called hubs [45], tend to be disproportionally nearest neighbors of many other nodes in the other graph. Direct nearest-neighbor rule favors these hubs, while at most, one of all the nearby nodes around a hub is a true anchor node. Thus, to diminish the hubness problem and identify nodes that are more isolated and more trustworthy to align with other nodes, we introduce the locally isolated similarity index (LISI).

**Locally isolated similarity index.** Due to the properties of translation invariance and scale invariance, we use Pearson correlation coefficient to measure the similarity between the embeddings of two nodes \( h_s \in H_s \) and \( h_t \in H_t \):

\[
corr(h_s, h_t) = \frac{\sum_i [h_s(i) - \bar{h}_s][h_t(i) - \bar{h}_t]}{\sqrt{\sum_i [h_s(i) - \bar{h}_s]^2} \sum_i [h_t(i) - \bar{h}_t]^2}. \tag{9}
\]

where \( \bar{h}_s, \bar{h}_t \) is the mean value of \( h_s \) and \( h_t \).

We consider a bi-partite neighborhood graph, in which each node of a given graph is connected to its \( m \) nearest neighbors in the other graph. Using \( N_i(h_s) \) to denote the neighborhood of node embedding \( h_s \) on target graph space, the hubness degree of node \( h_s \) is measured by computing its mean similarity to its target neighborhood as follow:

\[
D_i(h_s) = \frac{\sum_{h_t \in N_i(h_s)} \text{corr}(h_s, h_t)}{m}. \tag{10}
\]

By Eq. (10), the larger \( D_i(h_s) \) is, the more likely \( h_s \) is to be a hub. Similarly, denoted by \( D_s(h_t) \), the mean similarity between \( h_t \) and its source neighborhood can be computed. They are used to calculate the similarity measure LISI:

\[
\text{LISI}(h_s, h_t) = 2 \cdot \text{corr}(h_s, h_t) - D_s(h_s) - D_t(h_t). \tag{11}
\]

Here, a bigger value of LISI can be obtained when these two embeddings are similar enough to each other (the first item is large) while they are as less similar as possible to their neighbors in the other graph (the latter two items are small).
In other words, LISI aims to identify two similar embeddings that are locally isolated rather than being hubs of dense areas.

**Trusted pairs.** The LISI values of node pairs across source graph and target graph are computed as the alignment matrix $M$. If two nodes are each other’s nearest neighbors with respect to LISI, they are regarded as a trusted pair. Mathematically, a trusted pair $(h_t, h_s)$ satisfies the following conditions:

$$
\begin{align*}
    h_t &= \arg\max_{h \in H_t} \text{LISI}(h_t, h), \\
    h_s &= \arg\max_{h \in H_s} \text{LISI}(h_t, h).
\end{align*}
$$

**Aggregation coefficient fine-tuning.** According to topology consistency, if two nodes are connected in a network, then their anchor nodes (if both exist) are likely to maintain connectivity in another network. In other words, if node $A$ and node $a$ form an anchor link across source graph and target graph, then potential anchor links are likely to exist across the neighborhood of node $A$ and that of node $a$. To find them, we have the following proposition.

**Proposition 2.** By assigning bigger aggregation coefficients to trusted (or known) anchor nodes, more potential anchor nodes around them can be identified.

**Proof.** Without loss of generality, suppose that there exist four nodes: $u, v \in V_s$ and $u', v' \in V_t$, where $(v, v')$ is a pair of trusted (or known) anchor nodes, $(u, u')$ is a pair of potential anchor nodes that needed to be identified and $v \in N_s(u), u' \in N_t(u')$. The first-layer embeddings of node $u$ and $u'$ can be expanded as:

$$
\begin{align*}
    H_{1,s,k}(u) &= f^0\left(\sum_{r \in N_s(u)} \tilde{L}_{s,k}(u, r) X_s(r) W^0\right) \\
    &= f^0\left([\tilde{L}_{s,k}(u, v) X_s(v) + \sum_{r \in N_s(u)} \tilde{L}_{s,k}(u, r) X_s(r)] W^0\right), \\
    H_{1,t,k}(u') &= f^0\left(\sum_{r' \in N_t(u')} \tilde{L}_{t,k}(u', r') X_t(r') W^0\right) \\
    &= f^0\left([\tilde{L}_{t,k}(u', v') X_t(v') + \sum_{r' \in N_t(u')} \tilde{L}_{t,k}(u', r') X_t(r')] W^0\right).
\end{align*}
$$

To increase the similarity between $H_{1,s,k}(u)$ and $H_{1,t,k}(u')$, we should make the terms in square brackets as close as possible. Note that $\tilde{L}_{s,k}(u, \cdot)$ and $\tilde{L}_{t,k}(u', \cdot)$ are scalars, which are generally regarded as aggregating weights, and $X_s(\cdot)$ and $X_t(\cdot)$ are vectors. In square brackets, the first item is a vector representing trusted anchor nodes, while the other item is a synthesized vector to aggregate the other nodes. As $(v, v')$ is a pair of trusted anchor nodes, $X_s(v) = X_t(v')$ holds due to attribute consistency. By enlarging the common vector components, namely the trusted-pair-related item, the synthesized vectors are tuned to be closer. In practice, since we do not know which pair of nodes around trusted pairs may have a potential anchor link, $\tilde{L}_{s,k}(\cdot, v)$ and $\tilde{L}_{t,k}(\cdot, v')$ should be tuned synchronously such that potential anchor links can be found and existing anchor links can be preserved. The process is similar for the following layers, which is omitted.

**Algorithm 2** TRUSTED-PAIR BASED FINE-TUNING

**Input:** Embeddings of source graph and target graph $H_s, H_t$, reinforcement rate $\beta$.

**Output:** Refined alignment matrices $M = \{M_0, M_1, \ldots, M_K\}$; the maximal number of trusted pairs $T_{max} = \{T_{max,0}, T_{max,1}, \ldots, T_{max,K}\}$.

1: Initialize reinforcement vectors $r_s, r_t$ with all-one vectors, number of trusted pairs $T_{cur} = \{T_{cur,0}, T_{cur,1}, \ldots, T_{cur,K}\}$; $T_{max}$ with all-zero vectors;
2: for $k = 0 \rightarrow K$ do
3: repeat
4: $T_{mk} \leftarrow T_{mk}$;
5: Compute LISI as alignment matrix $M_k$ using Eq. (9)-(11);
6: Identify and count trusted pairs $T_{mk}$ from $M_k$ using Eq. (12);
7: Update $r_s, r_t$ using Eq. (13);
8: Generate new embeddings $H_{s,k}, H_{t,k}$ using Eq. (14);
9: until $T_{mk} \leq T_{mk}$
10: end for
11: return $M, T_{max}$

**E. Posterior Importance Assignment**

After refinement procedure is done on each orbit, corresponding alignment matrices $M = \{M_0, M_1, \ldots, M_K\}$ are obtained. Each of them contains the alignment score of all node pairs, focusing on a specific order of topological consistency. To integrate those scores and produce the final
alignment matrix, we consider an alignment-related index: the number of trusted pairs identified in different orders of topological consistency. The motivation is that the more trusted nodes that can be identified, the higher the corresponding embedding quality.

We denote the weight of $M_k$ by $\gamma_k$, which is defined as:

$$\gamma_k = \frac{T_k}{\sum_i T_i},$$

where $T_i$ denotes the number of trusted pairs identified based on orbit $i$. Then the final alignment matrix is the weighted sum of each orbit-specific matrix: $M = \sum_k \gamma_k \cdot M_k$.

Given the final alignment matrix, for each node in source graph, the node with the highest score in target graph is regarded as the most possible anchor node.

### F. Complexity Analysis

Without loss of generality, we denote the number of nodes by $n$, the number of edges by $e$ (non-zero entries of adjacency matrix), the maximal node degree by $D$, and the dimension of hidden features by $d$.

**Time complexity.** There are mainly three steps to analyze:

- Orbit matrix construction. In this work, we use orbits in graphlets with not more than 4 nodes. Thanks to the scalable Orca algorithm [46, 47], edge orbit counting can be done in $O(eD^2)$ for 4-node graphlets, which allows us to count orbits on graphs with millions of edges within a second. And constructing orbit-based Laplacian matrices takes $O(e)$.

- Multi-orbit-aware embedding. The propagation of GCN encoder takes $O(Ked^L)$ where the numbers of orbits $K$ and hidden layers $L$ are usually small constants.

- Trusted-pair based fine-tuning. Identifying trusted-pairs takes $O(Kn^2)$, and refining the aggregation coefficients takes $O(n)$.

Therefore, the first two steps take $O(e)$ and the third step takes $O(n^2)$. Note that $O(n^2)$ is dominated by the global alignment task itself ($M \in \mathbb{R}^{n \times n}$), and most operations, such as trusted-pair identification, can be efficiently implemented through matrix manipulation. Also, considering the practical runtime shown in Fig. 7, the overall time cost of our approach is less than or comparable to that of most benchmark methods.

**Space complexity.** Three aspects need to be analyzed:

- Orbit-relevant storage. Due to the increasing sparsity of higher-order orbits, there are at most $O(e)$ non-zero entries in each orbit matrix. Therefore, the worst-case space complexity for storing $K$ sparse orbit Laplacian matrices is $O(Ke)$.

- Embedding-relevant storage. It takes $O(Ld^2)$ to store trainable parameters in $L$ GCN layers, and the space required for storing the features of $n$ nodes in $L$ hidden layers is $O(Lnd)$.

- Alignment-relevant storage. In the fine-tuning stage, the LISI matrix is computed for identifying trusted pairs and can be released from memory once the identification process is done. Hence, it takes $O(n^2)$ to store the LISI matrix in the whole refinement process. As for the weighted integration stage, a cumulative approach can be adopted, which only requires maintaining a main matrix and iteratively adding each orbit-specified score matrix, and thus it also takes $O(n^2)$, the same as most alignment methods.

In summary, the overall space complexity is $O(Ke + Ld^2 + Lnd + n^2)$. As the space complexity of a common GCN-based alignment approach would be $O(e + Ld^2 + Lnd + n^2)$, our method generally just requires extra space for storing sparse orbit matrices.

### V. Experimental Evaluations

In this section, we comprehensively evaluate the superiority of our unsupervised alignment framework on both real-world and synthetic datasets and compare it with state-of-the-art unsupervised and supervised methods. Other detailed evaluations are also reported which empirically demonstrate the effectiveness of the proposed method.

#### A. Experiment Setup

**Datasets.** To facilitate a fair comparison, in this work, we evaluate our approach by following the setup of the primary baselines [10], [13], [17] and using the same datasets as they do, including three widely used real-world benchmark datasets and two synthetic datasets. For synthetic datasets, the source networks are collected from the real world, while the target networks are generated from source networks in the same way as [10] by randomly removing a certain percentage of the edges in original networks. Node identity is preserved, indicating the alignment groundtruth. The statistics of these datasets are listed in Table I.

- **Allmovie & Imdb (MV):** each node is a movie and each edge between two movies signifies that they have at least one common actor. The former is constructed from Rotten Tomatoes website\(^1\) and the latter from Imdb website\(^2\). There are totally 5176 anchor links across these two networks [10].
- **Douban Online & Douban Offline (DB):** a pair of Chinese social networks, whose nodes denote users and edges represent whether two users are contacts or friends. This pair of networks contains 1118 anchor links [10], [13], [17].
- **Flickr & Myspace (FM):** Two subnetworks and their corresponding groundtruths are extracted by [17]. Each node represents a user, and part of the user’s profile information is treated as attributes. There are 267 anchor links across these two networks [13], [17].
- **Econ:** this dataset comes from an economic model of Victoria State, Australia during the banking crisis in 1880 [10], [48]. Each node is an organisation (like banks, firms), and the edges denote the contractual relationships between them. Partial edges (from 10% to 50%) are randomly removed to construct the target network.
- **BN:** this network represents a part of brain structure. The nodes are brain voxels, and each edge is a fiber tract that

\(^1\)https://www.kaggle.com/ayushkalla1/rotten-tomatoes-movie-database
\(^2\)https://www.kaggle.com/yotish1706/ imdbmoviesdataset
connects two voxels [10], [49]. Partial edges (from 10% to 50%) are randomly removed to construct the target network.

**Configurations for proposed framework.** If not stated otherwise, the hyperparameters are set as follows: number of GCN layers \( L = 2 \), embedding dimension \( d = 200 \), learning rate \( \eta = 0.01 \), number of nearest neighbors \( m = 20 \), aggregation reinforcement rate \( \beta = 1.1 \). The sensitivity of key hyperparameters is studied in V-F.

**Baseline methods.** Other six representative methods are constructed for comparison, including methods that only use topology information (IsoRank [14], PALE [18], CENALP [24]) and three methods that use both topology information and node attributes (FINAL [17], REGAL [18], GAlign [10]). In particular, GAlign is the most related competitor, which is also an unsupervised network alignment method based on GCN. Since other general embedding techniques, like DeepWalk [37]/Node2Vec [36]/LINE [38], have been discussed and surpassed in papers of PALE/CENALP/REGAL, we compare with these alignment methods directly. According to the original settings of these baseline methods, some of them have to introduce supervised information. Therefore, 10% of ground truth is used to train PALE and CENALP and generate the prior alignment matrix for FINAL and IsoRank.

**Metrics.** Two numerical metrics are used to comprehensively evaluate and compare the effectiveness of different alignment approaches. The first metric is \( \text{precision} \@ q \) (also known as \( \text{accuracy} \@ q \) or \( \text{success} \@ q \)) [17]. It calculates the rate that a node’s true anchor node can be found in its top-\( q \) candidates. Mathematically, \( \text{precision} \@ q \) is defined as:

\[
\text{precision} \@ q = \sum_{v_r \in V_s} \frac{\sum_{v_t \in \mathcal{V}_t(v_r)} (v_r^s, v_t) \in \mathcal{L}^s}{|\mathcal{L}^s|}
\]

where \( \mathcal{L}^s \) denotes the set of ground-truth anchor pairs, \( \mathcal{V}_t(v_r^s) \) denotes a subset of \( V_t \) which contains the target nodes that hold the top-\( q \) highest alignment scores in the row \( \mathbf{M}(v_r^s, \cdot) \), and \( 1(\cdot) \) is an indicator function.

Another metric, Mean Reciprocal Rank (MRR) [50], also known as Mean Average Precision (MAP) [19], evaluates the average performance of reciprocal rank. MRR is defined as:

\[
\text{MRR} = \frac{1}{|\mathcal{L}|} \sum_{i=1}^{\mathcal{|\mathcal{L}|}} \frac{1}{r_i}
\]

where \( r_i \) is the rank of \( i \)th true anchor node’s alignment score in the corresponding row of \( \mathbf{M} \). Note that for both \( \text{precision} \@ q \) and MRR, the higher the values are, the better the performance is. To mitigate randomness, all reported results are averaged over 20 runs.

**Computing infrastructures.** All experiments are conducted on a compute node with 32 Intel Xeon Gold 6242 CPUs (2.80GHz) and 2 Tesla V100-SXM2 GPUs (32GB of memory each). As for the main softwares, all methods are implemented based on NetworkX-2.6.0 and PyTorch-1.11.0 (if required).

**B. Overall Effectiveness**

The overall alignment performance is first investigated. We report the \( \text{precision} \@ 1 \), \( \text{precision} \@ 10 \), and MRR metrics for numerical comparison purpose, which are presented in Table II. As can be seen, the proposed HTC framework consistently outperforms all baseline models across three pairs of real-world datasets. Specifically, we achieve up to 33.93%, 2.63%, and 6.25% relative improvements over the best results achieved by baselines in terms of \( \text{precision} \@ 1 \), \( \text{precision} \@ 10 \), and MRR, respectively.

Although 10% of ground truth is provided for supervised methods, it seems inadequate to make them outstanding. Among them, FINAL is the best-performing supervised model, which achieves a competitive result in terms of \( \text{precision} \@ 10 \) on Allmovie & Imdb dataset. Even so, compared with it, our HTC is fully unsupervised, and is 10.31% more accurate on Allmovie & Imdb dataset in terms of \( \text{precision} \@ 1 \), which is the most important metric.

As for unsupervised competitors, GAlign is the most powerful one. Like our HTC, it does not focus solely on the original network topology. Instead, GAlign utilizes augmented networks to increase its adaptiveness to structural consistency violation, and thus it is the runner-up on three pairs of datasets in terms of several metrics. In addition to steadily outperforming GAlign in terms of accuracy, our method likewise exhibits robustness to structural noise, which is further discussed in subsection V-D.

It is worth noting that all alignment methods perform poorly on Flickr & Myspace datasets. In addition to very sparse network topology and very limited attribute information as listed in Table I, according to the insights provided in [13], another important factor that makes Flickr & Myspace difficult to align is that their groundtruths rarely satisfy the common alignment consistency that neighbours in the source network have connections in the target network. As a result, all methods can hardly achieve satisfactory performance, especially for the \( \text{precision} \@ 1 \) metric. However, our HTC is still the tallest among the short, and REGAL also shows relatively competitive results on this pair of datasets.

We also investigate the importance scores of different levels of orbit-weighted topology computed by Eq. 15 on three pairs of real-world datasets. In Fig. 6, we rank orbits by their \( \gamma \) values, and the higher the position of the orbit in the plot, the higher its \( \gamma \) value. Some interesting and insightful observations are as follows: (1) The overall distributions of orbit weights over three datasets are quite different. The weights on Allmovie & Imdb datasets have the smallest variance than that of the other two pairs of datasets. This is consistent with

| Networks   | #Edges | #Nodes | #Attrs | Avg. Deg |
|------------|--------|--------|--------|----------|
| Allmovie   | 124709 | 6011   | 14     | 41.4     |
| Imdb       | 119073 | 5713   | 14     | 41.7     |
| Douban Online | 8164  | 3906   | 538    | 4.2      |
| Douban Offline | 1511  | 1118   | 538    | 2.7      |
| Flickr     | 7333   | 6714   | 3      | 2.2      |
| Myspace    | 11081  | 10733  | 3      | 2.1      |
| Econ       | 7619   | 1258   | 20     | 12.1     |
| BN         | 9016   | 1781   | 20     | 10.1     |

**TABLE I: Statistical details of networks**

---

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
To comprehensively evaluate the efficiency of our proposed HTC, the runtime of HTC on three pairs of real-world datasets is reported in Table II. Besides, to facilitate comparison with the baseline methods, we present runtime results in Fig. 7 (CENALP takes much more time and is therefore excluded). As illustrated, HTC takes the least time to achieve the best accuracy on two out of three datasets (Douban online & offline and Flickr & Myspace), and the time on another dataset (Allmovie & Imdb) is also comparable to most benchmarks. The performance of our HTC is evaluated and compared with that of baselines, which is shown in Fig. 8. In particular, the time for other operations refers to the rest time consumption in addition to the other five parts of the total time, such as the time for calculating assessment metrics. On three datasets, orbit counting, Laplacian matrix construction, and weighted integration take relatively less time than the other processes. In addition to other operations, most of the time is spent on multi-orbit-aware training and trusted-pair based fine-tuning processes, which are the cores of our method.

In summary, taking both effectiveness and efficiency into consideration, our method shows its superiority over baselines.

### D. Robustness Test

To test the robustness of our proposed high-order topological consistency, in this subsection, we inject different levels of topological noise into synthetic datasets. Specifically, 10% to 50% of edges in original source networks are randomly removed to generate the corresponding target networks for Econ and BN datasets. The performance of our HTC is evaluated and compared with that of baselines, which is shown in Fig. 9. Generally, all methods suffer accuracy degradation as the noise level increases. Among those baselines, PALE and REGAL are more sensitive to structural noise than other baselines, while FINAL and CENALP are more robust. Besides, IsoRank performs less competitively under all different noise levels. In comparison, HTC consistently outperforms all baselines except GAAlign by a remarkable margin. Although our method does not target noise adaptation as GAAlign, it can still achieve better performance than GAAlign when the noise ratio is smaller than 40, which means redundant time consumption for extra running loops can be saved.

For an in-depth analysis, we divide the whole process of HTC into six parts, i.e., orbit counting, laplacian matrix construction, multi-orbit-aware training, trusted pair based fine-tuning, weighted integration, and other operations, and their corresponding time consumption is shown in Fig. 8. In particular, the time for other operations refers to the rest time consumption in addition to the other five parts of the total time, such as the time for calculating assessment metrics. On three datasets, orbit counting, Laplacian matrix construction, and weighted integration take relatively less time than the other processes. In addition to other operations, most of the time is spent on multi-orbit-aware training and trusted-pair based fine-tuning processes, which are the cores of our method.

In summary, taking both effectiveness and efficiency into consideration, our method shows its superiority over baselines.
than 0.3 (specifically, when ratio equals 0.1, the $p@1$ metrics of HTC\textsuperscript{HT} on Econ and BN are 0.9944/0.9781 and 0.9748/0.9524, respectively), and the degree of performance degradation is generally comparable to GAlign, which is far smaller than the other baselines like PALE and REGAL and demonstrates the robustness of our HTC. For example, the performance degradations of HTC, GAlign, and PALE on Econ dataset are 0.2448, 0.2213, and 0.4522, respectively.

We owe the robustness of HTC to our multi-orbit-aware training mechanism based on two facts. First, not all higher order orbits are present for a given edge, e.g., the counts of orbit 3 and 4 for edge $(a, b)$ are both 0 as shown in Fig. 5, which is equivalent to removing the edge from the corresponding orbit. Second, the encoder is shared and co-trained by all different orders of topological patterns. As a result, HTC is trained to be robust to structural noise, i.e., edge removal/missing.

E. Ablation Test.

To distinguish the contributions made by different components of HTC, we construct several variants and compare their performance on two real-world datasets (Because of the issues with Flickr & Myspace mentioned in subsection V-B, we exclude them from the ablation test and the subsequent hyperparameter study). All of them are built on top of a two-layer GCN with relative hyperparameters consistent with HTC.

- **HTC-L** (low-order, w/o fine-tuning): only considers low-order topology, namely the simplest pattern (orbit 0) to learn node features and compute alignment matrix without a refinement process.
- **HTC-H** (high-order, w/o fine-tuning): utilizes multi-orbit-aware training techniques to learn higher-order topological consistency while removes the refinement step.
- **HTC-LT** (low-order, w/ fine-tuning): learns node embeddings based on the simplest pattern, while the alignment matrix is computed after conducting trusted-pair based fine-tuning.
- **HTC-DT** (diffusion, w/ fine-tuning): replaces our GOMs with diffusion matrices of varying orders, which help to capture a larger neighborhood of structural information (best result is achieved with $k = 5$ and teleport probability $\alpha = 0.15$) [27].

The results are reported in Table III. Compared with the other four variants, HTC (namely HTC-HT) performs far better in terms of both $p@1$ and MRR metrics. Specifically, HTC achieves up to a 49% absolute improvement in $p@1$ over HTC-L, which ignores the proposed higher-order topological consistency and computes the alignment matrix directly without fine-tuning. This verifies the significant contributions we make to the original GCN in the context of unsupervised network alignment. Furthermore, HTC-H and HTC-LT outperform HTC-L by up to 38% and 16%, respectively, which indicates that more contributions are made by introducing higher-order topological consistency.

The diffusion matrix proves helpful for graph learning since multi-hop neighborhoods are considered instead of direct one-hop neighbors [27]. Thus, we take it as a representative technique that considers a larger range of neighboring topology based on the trivial edge pattern for aggregation information (HTC-DT). However, it performs even worse than HTC-L in the unsupervised alignment task. The fact is, GOMs can generally provide more sparse but more accurate topological consistency information when the order increases, while diffusion matrices are quite the opposite. Densified matrices make it harder to identify good anchor links because structural specificity is reduced and node embeddings become more smooth. Therefore, simply considering a larger range of neighboring topologies may hardly provide favorable information, especially in the unsupervised alignment scenario.

F. Hyperparameter Study

We study key hyperparameters of HTC here.

1. **The number of orbits ($K$).** Orbits have two properties. First, the number of induced subgraphs on $N$ nodes increases exponentially with $N$ and so does the number of orbits defined on $N$ nodes. Second, the larger a graphlet is, the less frequently it occurs since such a graphlet requires more nodes and edges to form a specific local topology. As a result, the orbit matrix of a higher-order pattern is generally more sparse, which means less consistency information. The trend of incrementally using first $K$ orbits is displayed in Fig. 10a. It can be found that increasing the number of orbits used at the beginning significantly improves the performance. The growth rate then decays, showing a smaller slope on the curve. The precision is no longer increasing and even slightly decreased when $K$ is larger than 13. Since considering too many orbits may bring little information gain (For instance, in Fig. 6, orbits 11, 12 account least for the model on both datasets) and cost more computation, in this work, we consider orbits in induced subgraphs with 2-4 nodes, i.e. the first 13 orbits.

2. **Structural hyperparameters of GCN ($L; \ d$).** It has been widely demonstrated that the best performance of a GCN model is achieved with 2 or 3 layers [10], [29], [39], and so
is our model (we achieve best performance with \( L = 2 \)). As for the embedding dimension, it can be seen in Fig. 10b that the performance improves significantly at first and then slows down when \( d \) grows from 5 to 200. Besides, the \( p@1 \) metric on Douban dataset decreases when \( d > 200 \), which results from an over-fitting problem. It should be avoided by setting the dimension too large, which increases computational load and may even degrade performance. Hence, we set \( d = 200 \).

(3) The number of nearest neighbors (m). The results of the controlled experiment are shown in Fig. 10c. Generally, both too small and too large values of \( m \) lead to less satisfactory results, and the best performances on both datasets are achieved within the range [15, 25]. Hence, \( m \) is set as 20.

(4) Reinforcement rate (\( \beta \)). It can be seen from Fig. 10d that the performance of HTC decreases as \( \beta \) increases. This reveals that a small increase in the aggregation coefficient in each epoch can better detect more trusted pairs. Hence, we empirically set \( \beta = 1.1 \).

G. Visualization Analysis

We use t-SNE technique [51] to visualize the embedding of ground truth anchor nodes before and after being aligned by HTC, as shown in Fig. 11. We randomly sample 150 nodes from Douban Online dataset and their corresponding anchor nodes from Douban Offline dataset, which are mapped from high-dimensional feature space to 2-dimensional space. Due to space limitations, only five orbits are investigated and visualized here. The embeddings of two graphs before alignment have more distinct distributions (the upper five subgraphs). Many source graph embeddings (blue points) cannot find target graph embeddings (red points) that are close enough to them. While after alignment (the lower five subgraphs), the embeddings of source graph and target graph show a more similar distribution, and most node embeddings of source network can be covered by that of target network. This indicates that more consistency information between source graph and target graph is successfully captured by HTC.

VI. CONCLUSIONS

In this paper, we propose a fully unsupervised network alignment framework, named HTC, to develop topological consistency from lower order to higher order. We define higher-order topological consistency based on the introduced graphlet orbit matrix. By elegantly integrating higher-order consistency into the information aggregation process of GCN, the task of network alignment is converted into the task of node embedding similarity measurement. Instead of relying on any single level of topology, we train our encoder to be multi-orbit-aware in the unsupervised encoder-decoder paradigm. To further improve the alignment quality, we introduce a trusted-pair based fine-tuning mechanism. Accordingly, all orders of alignment consistency are integrated to obtain a comprehensive evaluation of node correspondence. Extensive experimental results on benchmark datasets well demonstrate our superiority, such as effectiveness, efficiency, and robustness, over a wide variety of state-of-the-art baselines. Our work verifies the significance of higher-order topological consistency for unsupervised alignment tasks. It also provides a general roadmap for improving the alignment performance of GCN-based models, as our method is built on the most common GCN-based alignment setting.

ACKNOWLEDGMENT

Qingqiang Sun is supported by the scholarship of China Scholarship Council (No. 202006320051). Ying Zhang is supported by ARC DP210101393. Wenjie Zhang is supported by ARC Future Fellowship FT210100303.

REFERENCES

[1] Y. Dong, J. Tang, S. Wu, J. Tian, N. V. Chawla, J. Rao, and H. Cao, “Link prediction and recommendation across heterogeneous social networks,” in 2012 IEEE 12th International conference on data mining, pp. 181–190, IEEE, 2012.
[2] Y. Xiang, G. Zhang, S. Gu, and J. Cai, “Online multi-layer dictionary pair learning for visual classification,” Expert Systems with Applications, vol. 105, pp. 174–182, 2018.
[3] C.-Y. Li and S.-D. Lin, “Matching users and items across domains to improve the recommendation quality,” in Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 801–810, 2014.
[4] L. Liu, W. K. Cheung, X. Li, and L. Liao, “Aligning users across social networks using network embedding,” in IJCAI, pp. 1774–1780, 2016.
[49] 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.

[50] T. Iofciu, P. Fankhauser, F. Abel, and K. Bischoff, “Identifying users across social tagging systems,” in Fifth International AAAI Conference on Weblogs and Social Media, 2011.

[51] L. Van der Maaten and G. Hinton, “Visualizing data using t-sne,” Journal of machine learning research, vol. 9, no. 11, 2008.