CombAlign: Enhancing Model Expressiveness in Unsupervised Graph Alignment

Songyang Chen, Yu Liu, Lei Zou, Zexuan Wang, and Youfang Lin

***Abstract*—Unsupervised graph alignment finds the node correspondence between a pair of attributed graphs by only exploiting graph structure and node features. One category of recent studies first computes the node representation and then matches nodes with the largest embedding-based similarity, while the other category reduces the problem to optimal transport (OT) via Gromov-Wasserstein learning. However, it remains largely unexplored in the model expressiveness, as well as how theoretical expressivity impacts prediction accuracy. We investigate the model expressiveness from two aspects. First, we characterize the model's *discriminative power* in distinguishing matched and unmatched node pairs across two graphs. Second, we study the model's capability of guaranteeing *node matching properties* such as one-to-one matching and mutual alignment. Motivated by our theoretical analysis, we put forward a hybrid approach named `CombAlign` with stronger expressive power. Specifically, we enable cross-dimensional feature interaction for OT-based learning and propose an embedding-based method inspired by the Weisfeiler-Lehman test. We also apply non-uniform marginals obtained from the embedding-based modules to OT as priors for more expressiveness. Based on that, we propose a traditional algorithm-based refinement, which combines our OT and embedding-based predictions using the ensemble learning strategy and reduces the problem to maximum weight matching. With carefully designed edge weights, we ensure these matching properties and further enhance prediction accuracy. By extensive experiments, we demonstrate a significant improvement of 14.5% in alignment accuracy compared to state-of-the-art approaches and confirm the soundness of our theoretical analysis.**

***Index Terms*—Unsupervised graph alignment, model expressiveness, gromov-wasserstein discrepancy, graph learning.**

I. Introduction

THE unsupervised graph alignment problem predicts the node correspondence between two attributed graphs without any pre-aligned node pairs (i.e., anchors). It has a wide range of applications, including linking the same identity across different social networks [1], [2], matching scholar accounts between multiple academic platforms [3], [4], matching the protein networks from different species [5], [6], computing the graph edit distance between graphs [7], and various computer vision tasks [8], [9]. For instance, cross-species analysis of biomolecular networks finds the node matching between gene co-expression networks, protein-protein interaction networks, or metabolic networks in the unsupervised setting [10]. In real-world scenarios, given the substantial structural disparities among the nodes to be aligned, as well as the difficulty in obtaining pre-aligned node pairs, unsupervised graph alignment is both important and challenging.

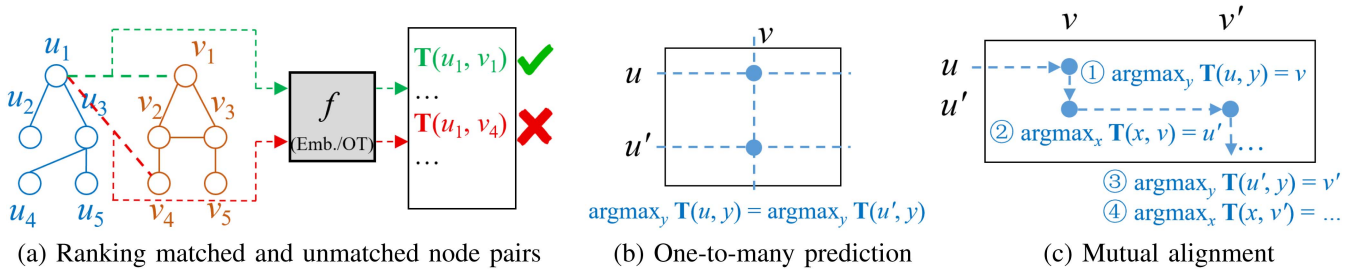
Significant research efforts have been dedicated to the unsupervised graph alignment problem. Early work [11], [12], [13] often formulates the task as the maximum common subgraph isomorphism or the quadratic assignment problem, of which the complexities are NP-hard. In recent years, learning-based methods have garnered increasing attention, surpassing traditional solutions in terms of alignment accuracy. One prominent category of them [14], [15], [16] relies on the *"embed-then-cross-compare"* paradigm. These methods first learn the embedding of each node in both graphs by exploiting the graph structure and node feature information, for example, with graph neural networks (GNNs). Then, two nodes are matched if their embeddings are close according to some specific metric such as cosine similarity [16]. However, for the unsupervised scenario, it is non-trivial to design an optimization objective that is fully suitable for the graph alignment task without any known node correspondence, which poses challenges for learning high-quality embeddings. The most recent work [16] employs a parameter-free approach without an explicit objective to achieve state-of-the-art performance. Another recent line of research [5], [17], [18], [19], [20] generally models the graph alignment problem via *optimal transport (OT)* and achieves promising accuracy due to its well-defined objective. Given the marginal distributions on two finite sets and a predefined transformation cost between them, the OT problem finds the joint distribution to minimize the total transport cost [21]. It has been shown that unsupervised graph alignment can be reduced to OT with the help of Gromov-Wasserstein (GW) discrepancy [5], [17], [22], which shares similar ideas with node alignment [5], [17]. For approaches of this paradigm, the crux lies in the definition of pairwise cost within each graph (i.e., *intra-graph* cost), then the transport cost (i.e., *inter-graph* cost) is computed accordingly. We have witnessed an increasingly sophisticated design of intra-graph cost [5], [17], [19] for better matching accuracy.

Received 7 April 2025; revised 15 October 2025; accepted 24 November 2025. Date of publication 4 December 2025; date of current version 30 December 2025. This work was supported by NSFC under Grant 62202037, Grant 62372031, and Grant 62372034. Recommended for acceptance by J. Tang. *(Corresponding author: Yu Liu.)*

Songyang Chen, Yu Liu, Zexuan Wang, and Youfang Lin are with the School of Computer Science and Technology, Beijing Jiaotong University, Beijing 100044, China, and also with the Beijing Key Laboratory of Traffic Data Mining and Embodied Intelligence, Beijing 100044, China (e-mail: songyangchen@bjtu.edu.cn; yul@bjtu.edu.cn; zexuanwang@bjtu.edu.cn; yflin@bjtu.edu.cn).

Lei Zou is with Peking University, Beijing 100871, China (e-mail: zoulei@pku.edu.cn).

Digital Object Identifier 10.1109/TKDE.2025.3640287



(a) Ranking matched and unmatched node pairs (b) One-to-many prediction (c) Mutual alignment

Fig. 1. Illustration of model expressiveness for graph alignment, where (a) denotes the model's capability of separating matched and unmatched node pairs, while (b) & (c) showcase two disadvantages of pure learning-based approaches, i.e., one-to-many prediction regardless of the one-to-one matching constraint, and the inconsistency between row and column-wise alignment.

Unfortunately, it remains largely unexplored for the *theoretical capability* of deep learning-based solutions.

Discriminative Power: We first investigate the model expressiveness for unsupervised graph alignment in the ability to distinguish matched and unmatched node pairs. As shown in Fig. 1(a) where u_i is assumed to align to v_i for $i \in \{1, \ldots, 5\}$, we see both embedding and optimal transport-based methods as learning a (parameterized) function f that maps two nodes in different graphs to their matching probability. We generalize existing approaches [5], [14], [16], [17] and focus on *ranking matched node pairs above the unmatched ones by the matching probabilities*. Please note that the model's accuracy is closely related to the discriminative power of f.

Node Matching Properties: We regret to find that two major disadvantages exist with pure learning-based approaches, which violate the property of graph alignment and deteriorate the accuracy. On the one hand, we note that the *one-to-one* matching constraint is forced in scenarios such as the alignment of the protein-protein interaction networks [23] because the conserved functional modules must be uniquely mapped. Although being the *de facto* setting for most studies [5], [15], [16], [17], [19], [24], [25], [26], [27], [28], [29], these methods typically produce one-to-many predictions (Fig. 1(b)) since they compute a matrix of matching probability for every pair of nodes and then take the row/column-wise maximum as the prediction. On the other hand, as aligned nodes are paired up in nature, the alignment is *mutual* (i.e., bidirectional). However, predicting the alignment directly via a probability-based approach might lead to inconsistency. As shown in Fig. 1(c), for node u, its matched node (i.e., v) should be aligned to another node u' according to the alignment probabilities. We tackle these disadvantages of pure learning-based approaches to enhance model's theoretical soundness and prediction accuracy.

Motivated by our viewpoint of model expressiveness, we improve the graph alignment process in the following aspects, and present a model framework named `CombAlign` to unify them in a principled way.

Enhancing the discriminative power of OT-based learning: First, we enhance the optimal transport-based paradigm [5], [17], [19], which has a well-defined objective, by incorporating *feature transformation* along with the well-adopted feature propagation [17], [19] in the transport cost design of GW learning. We prove that more discriminative power is obtained by enabling the cross-dimension mixture of features, which helps to distinguish matched and unmatched node pairs. As shown in Section V, with this single optimization, our model outperforms existing OT and embedding-based state-of-the-art.

Enhancing the discriminative power with WL: Second, instead of proposing complicated heuristics (e.g., [16]) under the embedding-based framework, we opt for a simple yet effective approach by using parameter-free GNNs to simulate the Weisfeiler-Lehman (WL) algorithm [30]. We also exploit embedding-based prior knowledge to improve the well-formed OT objective, more specifically, we compute *non-uniform marginals* for OT, a vital input to the learning process. The learning algorithm is proved to be more expressive by optimizing an orthogonal aspect to feature transformation.

Guaranteeing node matching properties with improved accuracy: Third, to solve the disadvantages in Fig. 1(b) and (c), we reduce graph alignment to the *maximum weight matching* problem [31], [32] by properly establishing a set of weighted edges between two node sets. To define the edge weights, which determine the matching accuracy, we combine the predictions of our OT and embedding-based procedures via an ensemble learning strategy [33], [34] named *stacking*. Our model not only guarantees the matching properties but also improves prediction accuracy by combining the best of both worlds.

Our contributions for the unsupervised graph alignment problem are summarized as follows:

- We combine optimal transport and embedding-based approaches to improve the model expressiveness in *distinguishing matched and unmatched node pairs*. In particular, we demonstrate the necessity of feature transformation and non-uniform marginals in the learning process.
- We transform unsupervised graph alignment into a maximum weight matching problem with an ensemble learning strategy to integrate embedding and OT-based predictions. This not only ensures important matching properties but also enhances prediction accuracy.
- We propose `CombAlign`, a unified framework for unsupervised graph alignment to fully integrate the advantages of embedding-based, OT-based, and traditional algorithm-based solutions.
- Extensive experiments are conducted to evaluate our algorithm against both embedding and OT-based state-of-the-art. Our method improves alignment accuracy by a significant margin, while a detailed evaluation shows the effectiveness of each proposed module.

II. Related Work

Unsupervised Graph Alignment: Classic graph alignment methods, such as `EigenAlign` [11], formulates the problem as a Quadratic Assignment Problem [12], which considers both matches and mismatches and solves it by spectral decomposition of matrices. Meanwhile, `LREA` [35] addresses the problem by solving a maximum weight bipartite matching problem on a low-rank approximation of a node similarity matrix, and `GRASP` [36] treats the problem as a special case of finding a mapping between functions on graphs with the linear assignment algorithm. Recently, the research focus has shifted towards learning-based methods, which predict an alignment matrix indicating the matching probability for every node pair. Existing unsupervised solutions can be roughly divided into two categories. The first category, referred to as *"embed-then-cross-compare"* [14], [15], [16], [37], [38], [39], tackles the problem by first generating node representation for both graphs, e.g., using GNNs. Then, the alignment probability is computed by specific similarity measures based on node embeddings, e.g., the cosine similarity [14], [16]. In particular, `GAlign` [14] incorporates the idea of data augmentation into the learning objective to obtain high-quality node embeddings. `GTCAlign` [16] further simplifies `GAlign` by using a GNN with randomized parameters for node embedding computation. Another recent work [15] conducts unsupervised graph alignment via the idea of generative adversarial networks (GANs).

Instead of designing sophisticated alignment rules based on node embeddings, the second category is known as *optimal transport (OT)*-based approaches. They share the well-defined objective of minimizing Gromov-Wasserstein discrepancy, which predicts the alignment probabilities given the transport cost between node pairs. These methods mainly differ in the specific form of the *learnable* transport cost. `GWL` [5] is the first work following this paradigm with predefined cost and incorporates Wasserstein discrepancy and reconstruction loss for regularization. As the state-of-the-art approach, `SLOTAlign` [17] extends `GWL` by multi-view structural modeling, which includes the linear combination of adjacency information, node features, and feature propagation. `UHOT-GM` [19] generalizes this idea with the notion of multiple and cross-modal alignment. The most recent work, `HLOT` [20], introduces hypergraph formulation to enrich inter-node relationships by modeling higher-order dependencies.

Semi-Supervised Graph Alignment: Unlike unsupervised graph alignment, with a set of anchor node pairs as input, the main idea of semi-supervised methods is to fully utilize the anchor information. A line of consistency-based algorithms [4], [6], [25], [40] are proposed to first define the neighborhood topology and attribute consistency followed by devising a procedure to propagate the anchor information across graphs, e.g., via Random Walk with Restart (RWR). Follow-up works resort to embedding-based approaches [26], [27], [28], [29], [41], [42], [43], [44], i.e., by learning node embeddings to make the anchor pairs close while preserving the graph structure in the latent space. Recently, several OT-based methods have attracted considerable attention. Ref. [24] leverages optimal transport to model the problem through alignment consistency, while [45] employs adaptive optimal transport by designing a learnable cost matrix, thereby achieving improved accuracy.

TABLE I
Table of Notations

Notation	Description
$\mathcal{G}_s, \mathcal{G}_t$	The source and target graphs
n, m	The sizes of the nodes and edges
$\mathbf{A}_p, \mathbf{X}_p$	Adjacency matrix, node feature matrix ($p = s, t$)
u_i, u_j, v_k, v_l	Nodes with $u_i, u_j \in \mathcal{G}_s$ and $v_k, v_l \in \mathcal{G}_t$
$\mathbf{Z}_p, \mathbf{H}_p, \mathbf{R}_p$	Embeddings computed by GNN, GNN w/o training, and feature propagation, respectively ($p = s, t$)
$\mathbf{C}_s, \mathbf{C}_t, \mathbf{C}_{\text{gwd}}$	Intra-graph and inter-graph costs for GW learning
$\mathbf{T}_{\text{WL}}, \mathbf{T}_{\text{GW}}$	The alignment probability matrices
$f(\cdot), g(\cdot)$	Functions with and without learnable parameters

Knowledge Graph Entity Alignment: A similar problem has been extensively studied [46], [47], [48], which aligns the entities across knowledge graphs [49]. For this problem setting, both topological structures and feature semantics are important. Building upon optimal transport, `FGWEA` [50] proposes a fused Gromov-Wasserstein framework for unsupervised knowledge graph entity alignment, which effectively integrates structural and semantic information to achieve robust performance. Recent advances [51], [52], [53] have explored leveraging large language models (LLMs) to enhance entity alignment with richer textual information from diverse corpora. Considering their different problem inputs, we regard recent LLM-based work as a distinct line of research.

III. Preliminary

Problem Statement: We denote an undirected and attributed graph as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ with node set $\mathcal{V}$ of size n, edge set $\mathcal{E}$ represented by the adjacency matrix $\mathbf{A} \in \{0,1\}^{n\times n}$, and $\mathbf{X} \in \mathbb{R}^{n\times d}$ corresponding to the node feature matrix with d-dimensional attribute vectors. We list the frequently used notations in Table I.

Definition 1 (Unsupervised Graph Alignment): Given source graph $\mathcal{G}_s$ and target graph $\mathcal{G}_t$, without any known anchor node pairs, unsupervised graph alignment returns a set of matched node pairs $\mathcal{M}$. For each node pair $(u_i, v_k) \in \mathcal{M}$, we have $u_i \in \mathcal{G}_s$ and $v_k \in \mathcal{G}_t$.

We assume that $n_1 = |\mathcal{V}_s|, n_2 = |\mathcal{V}_t|$, and $n_1 \le n_2$ w.l.o.g. In particular, we focus on the setting of *one-to-one* node alignment, following most existing studies [5], [14], [15], [16], [17], [19], [24], [26]. That is, every node can appear at most once in $\mathcal{M}$. Instead of directly computing $\mathcal{M}$, existing learning-based approaches [5], [14], [15], [16], [17], [19] predict an *alignment probability matrix* $\mathbf{T}$ of size $n_1 \times n_2$, where $\mathbf{T}(i,k)$ denotes the probability that $u_i \in \mathcal{G}_s$ is matched to $v_k \in \mathcal{G}_t$. Next, it is sufficient to set $\mathcal{M}(u_i) = \arg\max_k \mathbf{T}(i,k)$. Similar to existing studies [14], [15], [16], [17], [19], this works primarily focuses on attributed graphs.

Gromov-Wasserstein (GW) Learning: The discrete form of the Gromov-Wasserstein discrepancy is defined as follows.

Definition 2 (Gromov-Wasserstein Discrepancy (GWD) [5]): Given the distribution $\boldsymbol{\mu}$ (resp. $\boldsymbol{\nu}$) over $\mathcal{V}_s$ (resp. $\mathcal{V}_t$), the GW

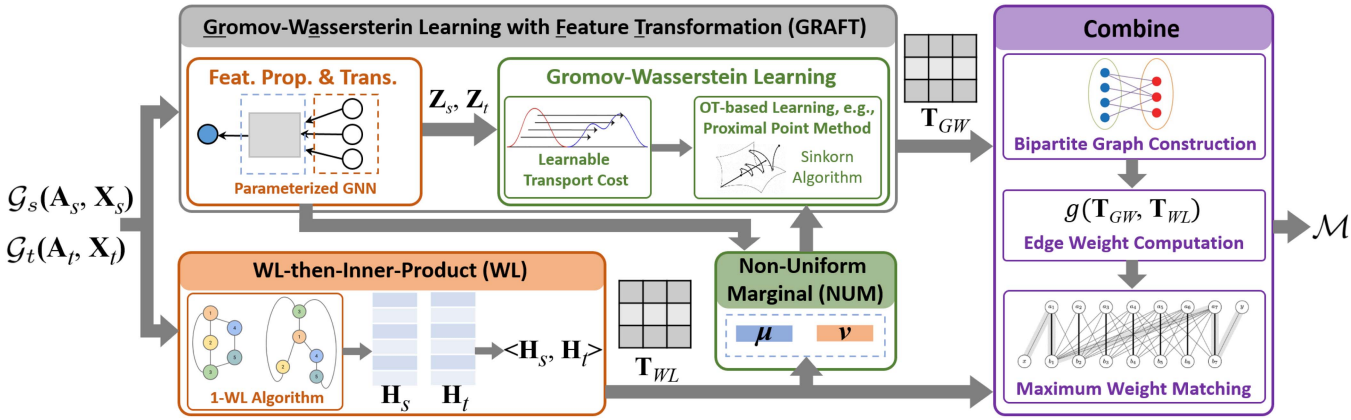


Fig. 2. The overall framework of CombAlign. Modules in orange, green, and purple belong to the embedding-based, OT-based, and traditional algorithm-based approaches, respectively.

discrepancy between $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ is defined as

$$\min_{\mathbf{T}\in\Pi(\boldsymbol{\mu},\boldsymbol{\nu})} \sum_{i=1}^{n_1}\sum_{j=1}^{n_1}\sum_{k=1}^{n_2}\sum_{l=1}^{n_2} |\mathbf{C}_s(i,j)-\mathbf{C}_t(k,l)|^2\mathbf{T}(i,k)\mathbf{T}(j,l),$$
$$\text{s.t.} \quad \mathbf{T}\mathbf{1}_{n_2}=\boldsymbol{\mu}, \mathbf{T}^\intercal\mathbf{1}_{n_1}=\boldsymbol{\nu}. \tag{1}$$

Here, $\mathbf{C}_s \in \mathbb{R}^{n_1\times n_1}$ and $\mathbf{C}_t \in \mathbb{R}^{n_2\times n_2}$ are the *intra-graph costs* for $\mathcal{G}_s$ and $\mathcal{G}_t$, respectively, which measure the similarity (or distance) of two nodes within each graph [22]. We have $\sum_{i=1}^{n_1}\sum_{k=1}^{n_2}\mathbf{T}(i,k)=1$ according to the constraints in Equation (1), i.e., $\mathbf{T}$ is the joint probability distribution over two node sets, and $\mathbf{1}_n$ denotes the all-ones vector in $\mathbb{R}^n$. Let $\mathbf{C}_{gwd}\in\mathbb{R}^{n_1\times n_2}$ be the *inter-graph cost matrix* [5], [22]:

$$\mathbf{C}_{gwd}(i,k)=\sum_{j=1}^{n_1}\sum_{l=1}^{n_2}|\mathbf{C}_s(i,j)-\mathbf{C}_t(k,l)|^2\mathbf{T}(j,l). \tag{2}$$

The above definition of $\mathbf{C}_{gwd}$ can be interpreted as follows. Noticing that Equation (1) can be reformulated as

$$\langle\mathbf{C}_{gwd},\mathbf{T}\rangle=\sum_{i=1}^{n_1}\sum_{k=1}^{n_2}\mathbf{C}_{gwd}(i,k)\mathbf{T}(i,k). \tag{3}$$

To minimize this objective with the constraints of $\mathbf{T}$, a negative correlation between the values of $\mathbf{C}_{gwd}(i,k)$ and $\mathbf{T}(i,k)$ is encouraged [21]. More precisely, for likely matched node pairs (u_i,v_k) and (u_j,v_l), $|\mathbf{C}_s(i,j)-\mathbf{C}_t(k,l)|^2$ should be small, i.e., the values of $\mathbf{C}_s(i,j)$ and $\mathbf{C}_t(k,l)$ are close [5], [17].

The Optimal Transport (OT) Problem: Actually, Equation (3) can be interpreted via optimal transport. Given $\mathbf{C}\in\mathbb{R}^{n_1\times n_2}$ which represents the cost of transforming probability distribution $\boldsymbol{\mu}$ to $\boldsymbol{\nu}$, we compute the joint probability distribution $\mathbf{T}\in\mathbb{R}^{n_1\times n_2}$ so that the total transport cost is minimized. It can be solved by the Sinkhorn algorithm [54], [55] via an iterative procedure. With learnable cost, existing solutions [5], [17], [19] adopt the proximal point method [56] to reduce GW learning to OT and to learn the alignment probability and parameters in the cost term jointly.

IV. Our Proposed Model

A. Model Overview

As shown in Fig. 2, given two graphs $\mathcal{G}_s$ and $\mathcal{G}_t$, CombAlign first predicts the node alignment by two separate approaches. The *WL-then-Inner-Product* (WL) module computes the embedding-based alignment probability $\mathbf{T}_{WL}$ by simulating the Weisfeiler-Lehman (WL) algorithm. Specifically, we employ parameter-free graph neural networks (GNNs) to derive node representations $\mathbf{H}_s$ and $\mathbf{H}_t$, and then compute their cosine similarity to obtain $\mathbf{T}_{WL}$. On the other hand, the *Gromov-Wasserstein Learning with Feature Transformation* (GRAFT) module computes an OT-based prediction $\mathbf{T}_{GW}$ with feature propagation & transformation followed by GW learning. It also interacts with the *Non-Uniform Marginal* (NUM) module to obtain the marginal distributions over the two node sets, which partially relies on the WL module. Given both predictions $\mathbf{T}_{WL}$ and $\mathbf{T}_{GW}$, the Combine module employs the maximum weight matching, a traditional algorithm with enhanced weight setting strategies, and returns the matched node pairs $\mathcal{M}$. Please note that without the Combine module, our model is end-to-end by using the alignment probability $\mathbf{T}_{GW}$ as prediction. In Algorithm 1, we list the overall process of CombAlign.

Our model framework design is backed by theoretical motivations. In the following, we formally demonstrate that the GRAFT module (more specifically, feature propagation & transformation) enhances the model's capability of ranking matched node pairs above the unmatched ones (Theorem IV.1 and Corollary IV.2). Our choice of non-uniform marginals (i.e., the NUM module) is supported by Theorem IV.3, improving model expressiveness from a different perspective. Finally, we guarantee the Combine module (with Theorem IV.4) to achieve the desired node matching properties, which also boosts alignment accuracy.

B. Improving Model's Discriminative Power

We elaborate on the designed modules by focusing on their discriminative power for graph alignment. We first introduce

Algorithm 1: The `CombAlign` Algorithm.

Input: Attributed graphs $\mathcal{G}_s(\mathbf{A}_s, \mathbf{X}_s)$ and $\mathcal{G}_t(\mathbf{A}_t, \mathbf{X}_t)$
Output: The set of predicted node matching $\mathcal{M}$

$\mathbf{T}_{WL} \leftarrow \texttt{WL}(\mathcal{G}_s, \mathcal{G}_t)$; // Algorithm 4
$\boldsymbol{\mu}, \boldsymbol{\nu} \leftarrow \texttt{NUM}(\mathbf{T}_{WL})$; // Algorithm 5
$\mathbf{T}_{GW} \leftarrow \texttt{GRAFT}(\mathcal{G}_s, \mathcal{G}_t, \boldsymbol{\mu}, \boldsymbol{\nu})$; // Algorithm 3
$\mathcal{M} \leftarrow \texttt{Combine}(\mathbf{T}_{WL}, \mathbf{T}_{GW})$; // Algorithm 6
return $\mathcal{M}$;

Algorithm 2: Feature Propagation and Transformation (`FeatProp&Trans`).

Input: $\mathbf{A}, \mathbf{X}$, model parameter $\{\mathbf{W}^{(1)}, \ldots, \mathbf{W}^{(K)}\}$
Output: Node representation $\mathbf{Z}$

$\mathbf{Z}^{(0)} \leftarrow \mathbf{X}$;
for $k = 1$ *to* K **do**
  $\mathbf{Z}^{(k)} \leftarrow \sigma(g_{prop}(\mathbf{A}, \mathbf{Z}^{(k-1)}) \cdot \mathbf{W}^{(k)})$;
return $\mathbf{Z} \leftarrow \sum_{k=1}^{K} \mathbf{Z}^{(k)}$;

the `GRAFT` module followed by the `WL` and `NUM` modules as the latter two further enhance the expressiveness of the former.

1) The GW Learning (`GRAFT`) Module: It comprises embedding-based feature propagation and transformation followed by the GW learning step with the idea of OT.

Feature Propagation and Transformation: As pointed out by [17], for OT-based approaches, the alignment quality heavily depends on the specific design of intra-graph costs $\mathbf{C}_s$ and $\mathbf{C}_t$. Representative methods [5], [17], [19] integrate graph structures and node features as the following equation:

$$\mathbf{R}_p = g_{prop}(\mathbf{A}_p)\mathbf{X}_p, p = s, t, \tag{4}$$

where $g_{prop}(\cdot)$ is a general function without learnable parameters. For example, we can set it to the standard propagation of GCN [57], i.e., $g_{prop}(\mathbf{A}) = \tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}$ with $\tilde{\mathbf{D}} = \tilde{\mathbf{A}}\mathbf{1}$ and $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$. We use $\mathbf{R}_s \in \mathbb{R}^{n_1 \times d}$ and $\mathbf{R}_t \in \mathbb{R}^{n_2 \times d}$ to denote the result embedding matrices and refer to the process as *feature propagation*.[1]

We observe that this step only enables the propagation of features between a node and its neighbors in a *dimension-by-dimension* manner. However, interaction across different feature dimensions is not considered, which is also important, for example, when two feature dimensions share similar semantics. We put forward feature transformation, which is widely adopted by classical GNNs [57], [58], [59] and is formulated as

$$\mathbf{Z}_p = f_{GNN}(\mathbf{A}_p, \mathbf{X}_p, \mathbf{W}), p = s, t. \tag{5}$$

Here, f_{GNN} is a learnable function, and $\mathbf{W}$ denotes the learnable transformation matrices and is shared by two graphs, which enables feature interaction *across dimensions*. The following theorem demonstrates that, within the GW learning framework, the adoption of a single transformation layer for feature interaction enhances the discriminative power of the intra-graph cost matrices. Refer to all proofs in the Appendix.

Theorem IV.1: We are given the graph structures $\mathbf{A}_s, \mathbf{A}_t$ and node features $\mathbf{X}_s, \mathbf{X}_t$ as input. Denote feature propagation as $\mathbf{R}_p = g(\mathbf{A}_p)\mathbf{X}_p, p = s, t$, where $g(\cdot)$ is a function without learnable parameters. Denote the additional linear transformation as $\mathbf{Z}_p = \mathbf{R}_p\mathbf{W}$, where $\mathbf{W} \in \mathbb{R}^{d \times d}$ is the learnable matrix. Assume that we set the intra-graph cost matrices as $\mathbf{C}'_p = \mathbf{R}_p\mathbf{R}_p^\intercal$ and $\mathbf{C}_p = \mathbf{Z}_p\mathbf{Z}_p^\intercal$ for $p = s, t$, respectively, and let $(u_i, v_k), (u_j, v_l) \in \mathcal{M}^*$ and $(u_{j'}, v_l) \notin \mathcal{M}^*$ where $\mathcal{M}^*$ is the ground truth. Then, there exists a case that $|\mathbf{C}'_s(i,j) - \mathbf{C}'_t(k,l)| = |\mathbf{C}'_s(i,j') - \mathbf{C}'_t(k,l)|$ and $|\mathbf{C}_s(i,j) - \mathbf{C}_t(k,l)| \neq |\mathbf{C}_s(i,j') - \mathbf{C}_t(k,l)|$.

Moreover, we show that the discriminative power of our problem is closely related to that of GNNs (e.g., [59]).

Corollary IV.2: Given two GNNs f_{GNN} and f'_{GNN} for feature propagation & transformation, where the expressive capability of f_{GNN} is strictly more powerful than that of f'_{GNN}. Denote by $\mathbf{C}$ and $\mathbf{C}'$ the intra-graph costs derived from f_{GNN} and f'_{GNN}, respectively. Then, our method has more discriminative power of ranking matched node pairs above the unmatched ones with $\mathbf{C}$.

Motivated by the theoretical results, we employ three specific GNN models for feature propagation and transformation:

- Lightweight GCN [60] (single linear transformation):

$$\mathbf{Z} = \text{Concat}(\mathbf{X}, \mathbf{P}\mathbf{X}, \ldots, \mathbf{P}^K\mathbf{X})\mathbf{W}, \mathbf{P} = \tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}. \tag{6}$$

- Graph Convolutional Network (GCN) [57]:

$$\mathbf{Z}^{(k+1)} = \sigma(\mathbf{P}\mathbf{Z}^{(k)}\mathbf{W}^{(k+1)}), k \in [0, K-1]. \tag{7}$$

- Graph Isomorphism Network (GIN) [59]:

$$\mathbf{Z}^{(k+1)} = \text{MLP}^{(k+1)}((1 + \varepsilon^{(k+1)})\mathbf{Z}^{(k)} + \mathbf{A}\mathbf{Z}^{(k)}), k \in [0, K-1]. \tag{8}$$

For GCN and GIN, we use the skip connection to combine node embeddings of all layers: $\mathbf{Z}_p = \sum_{k=1}^{K} \mathbf{Z}_p^{(k)}, p = s, t$. The pseudocode is shown in Algorithm 2.

Gromov-Wasserstein Learning: Given the cost matrices as input, we follow the well-adopted approach in [5], [17], [19] by employing the proximal point method [5], [56] and reducing the learning problem to optimal transport [54]. Denote by $L(\mathbf{T}, \boldsymbol{\beta}, \mathcal{W})$ the learning objective in Equation (3), where $\boldsymbol{\beta} = (\boldsymbol{\beta}_s, \boldsymbol{\beta}_t)$ is the learnable coefficients to combine multiple terms for the intra-graph cost and $\mathcal{W}$ denotes the parameters in feature transformation. We update $\mathbf{T}$ and $\boldsymbol{\Theta} = \{\boldsymbol{\beta}, \mathcal{W}\}$ by the proximal point method and gradient descent, respectively.

$$\boldsymbol{\Theta}^{(i+1)} = \arg\min \left\{ \nabla_{\boldsymbol{\Theta}} L(\mathbf{T}^{(i)}, \boldsymbol{\beta}^{(i)}, \boldsymbol{\Theta}^{(i)})^\intercal \boldsymbol{\Theta} + \frac{1}{2\tau_\Theta} \|\boldsymbol{\Theta} - \boldsymbol{\Theta}^{(i)}\|^2 \right\},$$

$$\mathbf{T}^{(i+1)} = \arg\min \left\{ \nabla_{\mathbf{T}} L(\mathbf{T}^{(i)}, \boldsymbol{\beta}^{(i)}, \boldsymbol{\Theta}^{(i)})^\intercal \mathbf{T} + \frac{1}{\tau_T} \text{KL}(\mathbf{T} \| \mathbf{T}^{(i)}) \right\}. \tag{9}$$

[1] Note that [17] uses parameter-free GNN to denote this process. In this paper, we propose strict definitions for both terms and formally analyze their differences.

Algorithm 3: Unsupervised Gromov-Wasserstein Learning With Feature Transformation (GRAFT).

Input: $\mathcal{G}_s$ and $\mathcal{G}_t$, marginal distributions $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$
Output: The OT-based alignment probability $\mathbf{T}_{GW}$

1 $\mathbf{T}_{GW} \leftarrow \boldsymbol{\mu}\boldsymbol{\nu}^\intercal$, $\boldsymbol{\beta}_s, \boldsymbol{\beta}_t \leftarrow (1,1,1)^\intercal$;
2 **for** $i = 1$ *to* I **do**
3 $\quad \mathbf{Z}_s \leftarrow \texttt{FeatProp\&Trans}(\mathbf{A}_s, \mathbf{X}_s, \mathbf{W}^{(1,\dots,K)})$;
4 $\quad \mathbf{Z}_t \leftarrow \texttt{FeatProp\&Trans}(\mathbf{A}_t, \mathbf{X}_t, \mathbf{W}^{(1,\dots,K)})$;
$\quad$ // intra-graph cost (Eq. 15)
5 $\quad \mathbf{C}_p \leftarrow f_{\boldsymbol{\beta}}(\mathbf{A}_p, \mathbf{X}_p, \mathbf{Z}_p), p = s, t$;
$\quad$ // inter-graph cost (Eq. 1)
6 $\quad \mathbf{C}_{gwd} \leftarrow f_{gwd}(\mathbf{C}_s, \mathbf{C}_t, \mathbf{T}_{GW})$;
7 $\quad$ Minimize $\langle \mathbf{C}_{gwd}, \mathbf{T}_{GW} \rangle$ by updating $\boldsymbol{\Theta} = \{\boldsymbol{\beta}, \mathbf{W}^{(1,\dots,K)}\}$;
$\quad$ // the proximal point method
8 $\quad$ Initialize $\mathbf{T}^{(0)} \leftarrow \mathbf{T}_{GW}$ and $\mathbf{a} \leftarrow \boldsymbol{\mu}$;
9 $\quad$ **for** $i' = 0$ *to* $I_{ot} - 1$ **do**
10 $\quad\quad$ Set $\mathbf{G} \leftarrow \exp\left(-\frac{\mathbf{C}_{gwd}}{\tau_T}\right) \odot \mathbf{T}^{(i')}$;
$\quad\quad$ // Sinkhorn-Knopp algorithm
11 $\quad\quad$ **for** $j = 1$ *to* J **do**
12 $\quad\quad\quad$ $\mathbf{b} \leftarrow \frac{\boldsymbol{\nu}}{\mathbf{G}^\intercal \mathbf{a}}$;
13 $\quad\quad\quad$ $\mathbf{a} \leftarrow \frac{\boldsymbol{\mu}}{\mathbf{G}\mathbf{b}}$;
14 $\quad\quad$ $\mathbf{T}^{(i'+1)} \leftarrow \mathrm{Diag}(\mathbf{a})\mathbf{G}\mathrm{Diag}(\mathbf{b})$;
15 $\quad$ $\mathbf{T}_{GW} \leftarrow \mathbf{T}^{(I_{ot})}$;
16 **return** $\mathbf{T}_{GW}$;

Note that $\mathbf{KL}(\cdot||\cdot)$ is the Kullback-Leibler divergence, and $\tau_\Theta = (\tau_\beta, \tau_W)$ is the learning rates for updating $\boldsymbol{\beta}$ and $\mathcal{W}$, respectively, while τ_T is the regularization coefficient in the Sinkhorn algorithm. It can be proved that with the simplified version of feature transformation, e.g., by adopting the lightweight GCN, the GW learning procedure theoretically guarantees the convergence result as in [5], [17] (Cf. Appendix).

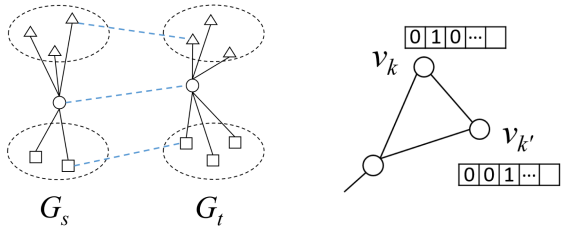
The pseudocode of the GRAFT module is demonstrated in Algorithm 3. We first initialize the OT-based alignment matrix $\mathbf{T}_{GW}$ with the input marginals and set the learnable coefficients (Line 1). Then, for each iteration (Line 2), we invoke the feature propagation and transformation process to derive node representations (Lines 3-4), based on which the intra and inter-graph costs are computed (Lines 5-6). The GW learning procedure is employed (Lines 7-16), which takes the Sinkhorn algorithm [55] as a subprocedure to iteratively update $\mathbf{T}_{GW}$, an $n_1 \times n_2$-sized alignment probability matrix.

2) The WL-Then-Inner-Product (WL) Module: The WL module uses a GNN with non-learnable parameters (referred to as *parameter-free GNNs*) to simulate the Weisfeiler-Lehman (WL) test [30]. As demonstrated in Algorithm 4, we use K random matrices $\widehat{\mathbf{W}}^{(1)}, \dots, \widehat{\mathbf{W}}^{(K)}$ to resemble the hash functions in WL test (Line 1). Taking node features as input, the algorithm conducts K layers of graph convolution (Lines 2-3), similar to the GNN in Section IV-B1 but with non-updated parameters. We denote by $\mathbf{H}_s$ and $\mathbf{H}_t$ the node embeddings in order to distinguish them from those obtained via a learnable GNN

Algorithm 4: WL-Then-Inner-Product (WL).

Input: $\mathcal{G}_s(\mathbf{A}_s, \mathbf{X}_s)$ and $\mathcal{G}_t(\mathbf{A}_t, \mathbf{X}_t)$
Output: Embedding-based alignment probability $\mathbf{T}_{WL}$

1: Initialize non-learnable parameters $\widehat{\mathbf{W}}^{(1)}, \dots, \widehat{\mathbf{W}}^{(K)}$;
2: $\mathbf{H}_s \leftarrow \texttt{FeatProp\&Trans}(\mathbf{A}_s, \mathbf{X}_s, \widehat{\mathbf{W}}^{(1,\dots,K)})$;
3: $\mathbf{H}_t \leftarrow \texttt{FeatProp\&Trans}(\mathbf{A}_t, \mathbf{X}_t, \widehat{\mathbf{W}}^{(1,\dots,K)})$;
4: $\mathbf{T}_{WL} \leftarrow \mathrm{Norm}(\mathbf{H}_s \mathbf{H}_t^\intercal)$;
5: **return** $\mathbf{T}_{WL}$;



(a) An illustrative example (b) An inseparable case

Fig. 3. Illustration of the necessity of non-uniform marginals.

(i.e., $\mathbf{Z}_s$ and $\mathbf{Z}_t$). The embedding-based alignment probability $\mathbf{T}_{WL}$ is obtained by multiplying $\mathbf{H}_s$ and $\mathbf{H}_t$, followed by the normalization process (Line 4). We first adopt a ReLU activation and then divide each term by the summation of all terms in $\mathbf{T}_{WL}$. In this way, each element of the returned $\mathbf{T}_{WL}$ denotes the matching probability.

Although the prediction of WL is not as precise as GRAFT due to its simplicity, as we will show in the following, it serves as the prior knowledge for the OT-based component to boost matching accuracy.

3) The Non-Uniform Marginal (NUM) Module: It improves the GRAFT module by providing non-uniform marginals as prior knowledge. We observe that OT-based approaches take two marginal distributions on $\mathcal{V}_s$ and $\mathcal{V}_t$ as input, which is commonly assumed to be uniformly distributed [5], [17], [19]. However, graphs are typical *non-Euclidean* data where node distributions are not i.i.d. A motivating example is shown in Fig. 3(a), where both $\mathcal{G}_s$ and $\mathcal{G}_t$ contain two clusters and a hub node, respectively. Suppose that the correspondence between nodes is consistent with their shapes. If the pair of hubs is correctly aligned, it might eliminate a large amount of impossible alignments. We formally show that non-uniform marginals indeed enhance the expressive power of GW learning for graph alignment with the following theorem.

Theorem IV.3: Consider the case that $(u_i, v_k) \in \mathcal{M}^*$ and $(u_i, v_{k'}) \notin \mathcal{M}^*$. For GW learning (e.g., GRAFT), under the mild assumption of the intra-graph cost, i.e.,

$$\mathbf{C}_s(i,i) = a, \forall u_i, \mathbf{C}_t(k,k) = b, \forall v_k, \tag{10}$$

$$\mathbf{C}_s(i,j) = \mathbf{C}_s(j,i), \mathbf{C}_t(k,l) = \mathbf{C}_t(l,k), \forall u_i, u_j, v_k, v_l, \tag{11}$$

$$\mathbf{C}_t(k,l) = \mathbf{C}_t(k',l), \forall v_l \in \mathcal{V}_t \backslash \{v_k, v_{k'}\}, \tag{12}$$

with uniform marginals $\boldsymbol{\mu} = (1/n_1, \dots, 1/n_1)^\intercal$ and $\boldsymbol{\nu} = (1/n_2, \dots, 1/n_2)^\intercal$, the first iteration of the GW learning process

Algorithm 5: Non-Uniform Marginal (NUM).

Input: Embedding-based alignment probability $\mathbf{T}_{WL}$
Output: Two marginal distributions $\boldsymbol{\mu}, \boldsymbol{\nu}$
1: $\boldsymbol{\mu} \leftarrow \mathbf{T}_{WL} \cdot \mathbf{1}_{n_2}, \boldsymbol{\nu} \leftarrow \mathbf{T}_{WL}^{\intercal} \cdot \mathbf{1}_{n_1}$;
2: **return** $\boldsymbol{\mu}, \boldsymbol{\nu}$;

with $\mathbf{T}^{(0)} = \boldsymbol{\mu}\boldsymbol{\nu}^{\intercal}$ cannot determine whether u_i is matched to v_k or $v_{k'}$.

Note that apart from feature transformation, which improves model's discriminative power via the first part of each term of $\mathbf{C}_{gwd}(i,k)$, i.e., $|\mathbf{C}_s(i,j) - \mathbf{C}_t(k,l)|^2$, we offer another perspective by investigating the second term $\mathbf{T}(j,l)$. To solve this problem, the following heuristics are provided to set non-uniform marginals without extra computational costs.

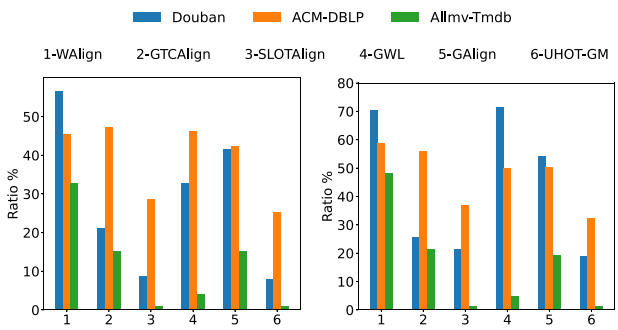
A fixed prior inspired by WL: We use the prior knowledge obtained from the WL test to set non-uniform marginals $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$. Given the embedding-based alignment probability $\mathbf{T}_{WL}$, the marginals are computed by row/column-wise summation (shown in Algorithm 5).

Adaptive marginals during GW Learning: Recall that $\mathbf{T}_{WL}$ is obtained by non-learnable embeddings $\mathbf{H}_s$ and $\mathbf{H}_t$, while the GRAFT module yields more informative node representation $\mathbf{Z}_s$ and $\mathbf{Z}_t$ through GW learning. Therefore, during each iteration (i.e., Line 2) of Algorithm 3, we use the learnable representations to compute an $n_1 \times n_2$-sized matrix following Algorithm 4, and take it as the input of Algorithm 5. The marginals are used to initialize the proximal point method (i.e., Lines 8, 12 & 13 of Algorithm 3) and are adjusted on the fly.

C. Ensuring Matching Properties (The Combine Module)

Motivation: As we have pointed out in Fig. 1(b) & (c) of Section I, pure learning-based methods [5], [16], [17], [19], [36] fail to guarantee alignment properties such as one-to-one matching and mutual alignment, albeit that they can achieve better practical accuracy than the traditional approaches [11], [12], [13]. In contrast, existing studies (e.g., [23]) have demonstrated real-world scenarios in which one-to-one node matching needs to be ensured. The inconsistency in mutual alignment means that for a predicted matching (u, v) taking from the row/column-wise maximum of the alignment probabilities, we have $\arg\max_y \mathbf{T}(u,y) = v$ and $\arg\max_x \mathbf{T}(x,v) \neq u$. This contradicts the essence of graph alignment where aligned nodes are paired up. We demonstrate in Fig. 4 that both problems are prevalent on real-world datasets. Given the output of an algorithm, we calculate the percentages of one-to-many predictions and inconsistencies in terms of mutual alignment, and find that a significant ratio of the predictions violates these alignment properties.

We observe that *maximum weight matching* is a feasible solution to this problem, which has been adopted by tradtional approaches. It takes a bipartite graph with weighted edges as input and finds a set of edges with maximum total weight on the condition that selected edges do not share any endpoint (i.e., node). By constructing a bipartite graph from the node sets, finding the maximum weight matching is equivalent to graph alignment with the additional benefit of holding the matching



(a) One-to-many prediction (b) Inconsistent alignment

Fig. 4. Percentage of one-to-many predictions and inconsistencies in terms of mutual alignment for representative baselines.

properties. The remaining challenge is to guarantee the accuracy of the alignment. Intuitively, the edge weights determine the alignment accuracy and should be close to the ground truth alignment. Our solution, referred to as the Combine module, tackles this problem by effectively combining both embedding and OT-based alignment probabilities. Specifically, it includes the following steps (Cf. Fig. 2).

Bipartite Graph Construction: To construct the bipartite graph $\mathcal{G}_b$, we set $\mathcal{V}_b = \mathcal{V}_s \cup \mathcal{V}_t$. Instead of building edges for all node pairs, we observe that in most cases, the ground truth is contained in the top-r predictions. As the GRAFT module generally achieves better performance, for each node $u_i \in \mathcal{G}_s$, we connect it to the nodes in $\mathcal{G}_t$ with top-r largest $\mathbf{T}_{GW}(i,\cdot)$. This step reduces the edge number from $n_1 n_2$ to $r n_1$ and significantly improves practical efficiency.

Edge Weight Computation: We adopt the idea of *ensemble learning* (more specifically, *stacking*) and see both embedding and OT-based predictions as base learners. We employ a simple yet effective parameter-free strategy to take the best of both worlds and believe that it tends to obtain more accurate predictions, i.e., the edge weight being close to the ground truth alignment. We consider the following concrete forms of edge weights and use the "both confident" setting by default:

- "Both confident": $g_{EL}(\mathbf{T}_{WL}, \mathbf{T}_{GW}) = \mathbf{T}_{WL} \odot \mathbf{T}_{GW}$,
- "Simple average": $g_{EL}(\mathbf{T}_{WL}, \mathbf{T}_{GW}) = (\mathbf{T}_{WL} + \mathbf{T}_{GW})/2$.

Maximum Weight Matching: Note that any maximum weight matching algorithm with real-value weights (e.g., the modified Jonker-Volgenant algorithm (MJV) [61]) can be invoked to solve the problem. The pseudocode of the Combine module is shown in Algorithm 6, which returns a set of (one-to-one) matched node pairs instead of the alignment probabilities.

Theorem IV.4: With the Combine module, our algorithm predicts a set of matched node pairs with desired properties, including one-to-one matching and mutual alignment.

D. Algorithm Analysis and Optimization

1) Complexity Analysis: We discuss the complexity of CombAlign by analyzing each module in brief. Denote by n, m the total number of nodes and edges in the graph, and d the hidden dimension. The WL module takes $O(K(md + nd^2) +$

Algorithm 6: The `Combine` Module.

Input: Embedding & OT-based outputs, $\mathbf{T}_{WL}, \mathbf{T}_{GW}$
Output: The combined prediction $\mathcal{M}$

Construct bipartite graph $\mathcal{G}_b$ with $\mathcal{V}_b \leftarrow \mathcal{V}_s \cup \mathcal{V}_t$;
Construct $\mathcal{E}_b$ by row-wise top-r prediction of $\mathbf{T}_{GW}$;
$\mathcal{E}_b.weight \leftarrow g_{EL}(\mathbf{T}_{WL}, \mathbf{T}_{GW})$;
Compute the maximum weight matching $\mathcal{M}$ for $\mathcal{G}_b$;
return $\mathcal{M}$;

TABLE II
COMPARISON OF MODEL COMPLEXITY WITH STATE OF THE ART

Method	Time Complexity
`GTCAlign` [16] (Emb.)	$O\big(IK(md+n^2d)\big) \approx O(n^2)$
`SLOTAlign` [17] (OT)	$O\big(I\big(K(md+nd^2)+n^2d+I_{ot}n^3\big)\big) \approx O(n^3)$
`CombAlign`	$O\big(I\big(K(md+nd^2)+n^2d+I_{ot}n^3\big)\big) \approx O(n^3)$
`CombAlign(Opt)`	$O\big(I\big(K(md+nd^2)+n^2(d+I_{ot}\log n)\big)\big) \approx O(n^2)$

$n^2\,d)$ time, where the first term follows classical message-passing GNNs and the second term correpsonds to the multiplication of two $n \times d$-sized embedding matrices. The `NUM` module only incurs $O(n^2)$ time for the matrix-vector multiplication. The `GRAFT` module has the same asymptotic complexity with `SLOTAlign` [17] since the additional feature transformation only needs $O(Kd^2)$ time. To be more specific, for both the `GRAFT` module and `SLOTAlign`, the cost computation of GW learning is $O(n^3)$ per round according to [5], with an inner-loop of I_{ot} rounds for the Proximal Point Method. For the `Combine` module, the bipartite graph construction step takes linear time, and the complexity is dominated by maximum weight matching. Although classical algorithms [31], [61], [62] typically need $O(n^3)$ time, note that our bipartite graph only contains $O(rn_1)$ edges, which corresponds to the *sparse* linear assignment problem with $O(n^2 r)$ complexity [63]. This process can be further improved with faster matching algorithms [64], [65]. In sum, the asymptotic complexity is $O(I(K(md+nd^2)+n^2\,d+I_{ot}n^3))$, which is in the same order as other optimal transport-based alignment methods [5], [17], [18], [19]. In comparison, we also demonstrate the time complexity of two representative methods in Table II. Note that both embedding and OT-based methods need to iteratively update the alignment probabilities, thus we use I to represent the number of iterations.

2) Optimizations: According to our analysis, the asymptotic complexity of `CombAlign` (and other OT-based approaches) can be simplified to $O(n^3)$ since n dominates other inputs (i.e., I, K, and d). Unfortunately, a time complexity gap is observed between embedding-based solutions [14], [15], [16] and `CombAlign`, while the former only takes roughly $O(n^2)$ time. The complexity bottleneck of OT arises from the GWD cost computation (i.e., $\mathbf{C}_{gwd}$). According to [5], it can be reformalized as $\mathbf{C}_{gwd} = \mathbf{C}_s^2\boldsymbol{\mu}\mathbf{1}_{n_2}^\intercal + \mathbf{1}_{n_1}\boldsymbol{\nu}^\intercal\mathbf{C}_t^2 - 2\mathbf{C}_s\mathbf{T}\mathbf{C}_t^\intercal$, reducing the computational complexity from $O(n^4)$ to $O(n^3)$. (Note that the square operation is element-wise.) Since only the third term has a complexity of $O(n^3)$, we simplify the intra-graph cost design by only utilizing the adjacency matrix, which leads to a $\mathbf{A}_s\mathbf{T}\mathbf{A}_t^\intercal$ term. For real-world graphs, the adjacency matrix is sparse, namely, the number of edges m is bounded by $O(n\log n)$ [66], and it holds that $m/n \ll d$ in practice (Cf. Section V). To this end, multiplying two sparse matrices with a dense one can be fulfilled in $O(nm) = O(n\log n)$ time. We also employ powerful embedding learning models (e.g., [67]) in the `GRAFT` module to alleviate the degradation of model expressiveness. As shown in Table II, our optimized model successfully closes the complexity gap between two categories of learning-based approaches.

For large-scale graphs, such as those with millions of entities, our algorithm can be easily integrated with a divide-and-conquer strategy [68]: first partitioning the source and target graphs into smaller subgraphs, then performing alignment among the subgraphs, and finally aggregating the matching results (Cf. Section V-B4).

TABLE III
DATASETS AND THEIR STATISTICS

Dataset	n_1, n_2	m_1, m_2	Features	Anchors
Douban [25]	1,118 3,906	3,022 16,328	538	1,118
ACM-DBLP [71]	9,872 9,916	39,561 44,808	17	6,325
Allmovie-Imdb [14]	6,011 5,713	124,709 119,073	14	5,174
Cora [72]	2,708 2,708	5,028 5,028	1,433	2,708
Citeseer [72]	3,327 3,327	4,732 4,732	3,703	3,327
PPI [73]	1,767 1,767	16,159 16,159	50	1,767
CS [74]	18,333 18,333	114,652 147,410	6,805	18,333
Physics [74]	34,493 34,493	347,147 446,332	8,415	34,493

V. EXPERIMENTS

We conduct a comprehensive experimental evaluation to answer the following key research questions:

- Q1. Does `CombAlign` outperform the state of the art?
- Q2. How effective is each component of `CombAlign`?
- Q3. Is `CombAlign` capable of holding matching properties such as one-to-one matching and mutual alignment?
- Q4. How efficient and scalable is `CombAlign`?

A. Experimental Settings

Datasets: We mainly use the well-adopted datasets for unsupervised graph alignment [5], [15], [16], [17], [19], including three real-world datasets Douban Online-Offline [25], ACM-DBLP [69], Allmovie-Imdb [14], and five synthetic ones, i.e., Cora [70], Citeseer [70], PPI [71], and two larger datasets CS and Physics [72]. The statistics of the datasets are listed in Table III, including the numbers of nodes (n_1 and n_2) and edges (m_1 and m_2), and known anchors only for evaluation. It is noteworthy that all ground-truth provided in Table III are one-to-one node correspondences. We adopt two datasets with one million nodes [68] for scalability analysis.

TABLE IV
COMPARISON OF MODEL PERFORMANCE ON SIX DATASETS

Datasets	Metrics	KNN	GAlign	WAlign	GTCAlign	GWL	SLOTAlign	UHOT-GM	HLOT	CombAlign w/o C
Douban	Hits@1	27.82	45.26	39.45	61.79	3.29	51.43	60.23	57.26	**68.52**
	Hits@5	45.53	67.71	62.35	76.83	8.32	73.43	71.36	79.77	**87.84**
	Hits@10	52.68	78.00	71.47	82.29	9.93	77.73	76.91	84.09	**91.41**
	MAP	36.08	56.32	46.22	69.77	5.79	61.29	67.35	63.45	**77.08**
ACM-DBLP	Hits@1	36.35	70.20	63.43	60.92	56.36	66.04	69.89	65.86	**72.18**
	Hits@5	66.83	87.23	83.18	75.60	77.09	85.84	87.12	84.74	**88.98**
	Hits@10	76.22	91.36	86.58	79.97	82.18	87.76	90.65	88.82	**92.63**
	MAP	50.11	77.49	70.76	67.67	64.82	73.76	77.18	72.89	**79.55**
Allmovie-Imdb	Hits@1	32.39	82.14	52.61	84.73	87.82	90.60	91.73	90.88	**96.25**
	Hits@5	51.57	86.35	70.91	89.89	92.31	92.75	94.36	93.01	**97.66**
	Hits@10	58.79	90.03	76.52	91.32	92.83	93.14	94.96	93.74	**97.89**
	MAP	41.50	84.96	61.17	87.12	89.64	91.61	92.74	91.83	**97.31**
Cora	Hits@1	95.01	99.45	98.45	99.35	86.19	99.48	99.48	99.27	**99.56**
	Hits@5	**100**	**100**	**100**	**100**	93.61	**100**	**100**	**100**	**100**
	Hits@10	**100**	**100**	**100**	**100**	94.57	**100**	**100**	**100**	**100**
	MAP	98.66	99.69	99.18	99.69	89.71	99.71	99.72	99.63	**99.75**
Citeseer	Hits@1	89.72	99.73	97.81	99.68	57.05	99.25	99.47	99.23	**99.82**
	Hits@5	**100**	**100**	**100**	**100**	65.04	**100**	**100**	**100**	**100**
	Hits@10	**100**	**100**	**100**	**100**	65.95	**100**	**100**	**100**	**100**
	MAP	94.91	99.84	98.88	99.89	61.31	99.62	99.69	99.79	**99.91**
PPI	Hits@1	84.96	89.20	88.51	89.25	86.76	89.30	89.33	89.26	**89.70**
	Hits@5	89.10	90.64	93.10	92.81	88.06	92.53	92.97	92.89	**93.15**
	Hits@10	92.17	94.16	94.17	94.07	88.62	93.49	93.52	93.46	**94.85**
	MAP	87.65	90.72	89.02	90.80	87.74	90.76	90.81	90.65	**91.12**

Baselines: We compare CombAlign with traditional solutions and representative embedding and OT-based models. For *traditional approaches*, we include KNN and use parameter-free GNNs to obtain node embeddings, based on which the top-k similar node pairs across graphs are matched. We also adopt the modified version of the Jonker-Volgenant algorithm [61] (MJV) for the linear assignment problem, which returns one-to-one alignment. We compare with representative *embedding-based models*, including WAlign [15], GAlign [14] and GTCalign [16], as well as *OT-based models*, i.e., GWL [5], SLOTAlign [17], UHOT-GM [19] and HLOT [20]. (Refer to Related Work for more details.) We omit other learning-based methods (e.g., [18], [36]) as they have been surpassed by the above baselines according to existing literature [5], [15], [16], [17].

Evaluation Metrics: Following previous studies [15], [16], [17], [19], we adopt Hits@k with $k = \{1, 5, 10\}$ and Mean Average Precision (MAP) to evaluate the model performance. Given the ground truth $\mathcal{M}^*$, for each $(u, v) \in \mathcal{M}^*$ with $u \in \mathcal{G}_s$ and $v \in \mathcal{G}_t$, we check if v belongs to the top-k predictions ordered by alignment probability, denoted as $S_k(u)$. Then, Hits@k is computed as follows:

$$\text{Hits@}k = \frac{\sum_{(u,v)\in\mathcal{M}^*} \mathbb{1}[v \in S_k(u)]}{|\mathcal{M}^*|}, \tag{13}$$

where $\mathbb{1}[\cdot]$ is the indicator function which equals 1 if the condition holds. MAP is employed to evaluate the accuracy of the predicted node rankings:

$$\text{MAP} = \frac{\sum_{(u_i,v_k)\in\mathcal{M}^*} \frac{1}{Rank(u_i,v_k)}}{|\mathcal{M}^*|}, \tag{14}$$

where $Rank(u_i, v_k)$ denotes the ranking of v_k according to $\mathbf{T}(i, \cdot)$ by descending order.

Implementation Details: For all baselines, we obtain the source code from the authors and make sure that all baseline models are tuned to the best performance. For datasets (e.g., Allmovie-Imdb) where some baselines do not use them in the original paper, we perform the hyperparameter search to obtain the best results. Our model is implemented based on PyTorch 1.12.1 and PyTorch Geometric. Specifically, for CombAlign, we set the feature dimension d as 32 and the graph convolution layers K as 3. The learning rates τ_β and τ_W for updating β and $\mathcal{W}$ are set to 1 and 0.01, respectively. We conduct experiments on a high-performance computing server with a GeForce RTX 4090 GPU equipped with 24 GB of memory.

B. Experimental Results

1) Model Performance: CombAlign **w/o the** Combine **module** (CombAlign w/o C). We first evaluate the prediction accuracy following previous learning-based settings [16], [17], specifically, the output is formulated as the alignment probability matrix. As shown in Table IV, we use the bold font to highlight the best results and underline the second-best values.[2] As for the baselines, note that GAlign achieves the best performance on ACM-DBLP while UHOT-GM has the highest accuracy on Allmovie-Imdb. For the Douban dataset, GTCAlign and UHOT-GM are the two best baselines. It can be concluded that neither OT-based methods nor embedding solutions can consistently outperform the other.

[2]The reported figures are the average of five independent rounds, and we omit the standard errors following most existing work because the prediction accuracy is highly concentrated for this problem.

TABLE V
COMPARISON OF HITS@1 WITH ONE-TO-ONE MATCHING CONSTRAINT

Datasets	Douban	ACM-DBLP	Allmovie-Imdb
KNN	23.88	31.11	28.79
MJV	31.03	56.89	35.14
GAlign	20.84	66.18	80.28
WAlign	15.56	60.33	51.29
GTCAlign	57.15	56.96	83.91
GWL	3.22	52.09	87.22
SLOTAlign	49.19	64.32	90.31
UHOT-GM	58.31	67.78	91.26
HLOT	54.69	64.14	90.37
CombAlign	**70.75**	**74.19**	**96.57**

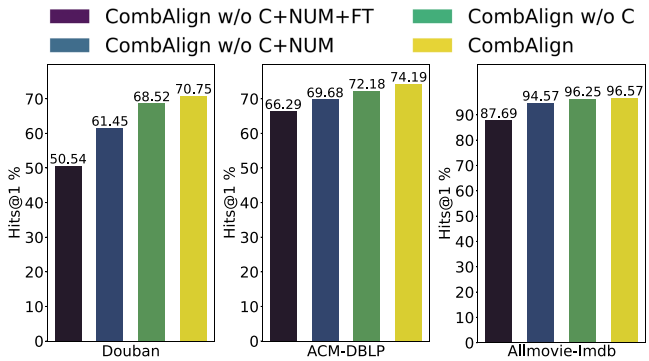


Fig. 5. Ablation study on three real-world datasets.

TABLE VI
THE PERFORMANCE (HITS@1) OF GRAFT WITH DIFFERENT GNNS

Hits@1	Douban	ACM-DBLP	Allmovie-Imdb
w/ LGCN	57.07	68.87	93.39
w/ GCN	61.45	69.68	94.57
w/ GIN	63.15	71.64	95.61
w/ SGFormer	65.38	72.73	96.02

For the embedding-based algorithms, we note that there is no exist a clear winner between GAlign and GTCAlign on the real-world datasets, demonstrating the limitation of the embedding-only heuristics in the unsupervised setting. Nonetheless, both of them outperform WAlign, while the training process of the latter encounters significant oscillations, partially due to the challenges in training GANs [73]. For the OT-based methods, in general, a more sophisticated design of intra-graph costs with multiple terms (i.e., SLOTAlign) and cross-modal comparison (i.e., UHOT-GM) result in better practical accuracy.

For the CombAlign w/o C algorithm, in our experiments, we adopt GCN [57] as the default GNN module in GRAFT and WL. It consistently demonstrates superior performance across all evaluation metrics on three real-world datasets, validating the effectiveness of our model. Notably, on the Douban and Allmovie-Imdb datasets, it achieves significant improvements in Hits@1, with relative gains of 10.89% and 4.93%, respectively. For the synthetic datasets, CombAlign w/o C also achieves the highest performance.

CombAlign **(w/ the** Combine **module)**. As presented in Table V, we evaluate CombAlign against the baselines by forcing the one-to-one matching constraint. Among the evaluated methods, only MJV and CombAlign inherently ensure one-to-one matching, as they directly output matched node pairs. For other methods generating the alignment probability matrices, we eliminate all one-to-many predictions by retaining the node pair with the highest probability. (Also note that mutual alignment is non-trivial to obtain for these methods.) It is observed that these methods inevitably suffer from performance degradation in this setting (Cf. Table IV), and the phenomenon is significant on Douban and ACM-DBLP datasets.

In comparison, CombAlign consistently outperforms all other methods. Notably, the substantial improvement of CombAlign over MJV underscores the critical role of weight design and demonstrates the effectiveness of the proposed ensemble learning strategy. Compared to CombAlign w/o C, CombAlign not only ensures one-to-one matching (and mutual alignment) but also significantly enhances performance.

2) Ablation Study: To test the effectiveness of each module, we progressively remove Combine, NUM, and feature transformation in GRAFT. The resulting model variants are as follows. For CombAlign w/o C, we make predictions by $\mathbf{T}_{GW}$ because it generally achieves higher accuracy by combining the prior knowledge from WL. CombAlign w/o C+NUM further eliminates the NUM and WL modules, therefore, the uniform marginal is adopted. The CombAlign w/o C+NUM+FT variant removes feature transformation as well as feature propagation, resulting in a model theoretically less expressive than SLOTAlign.

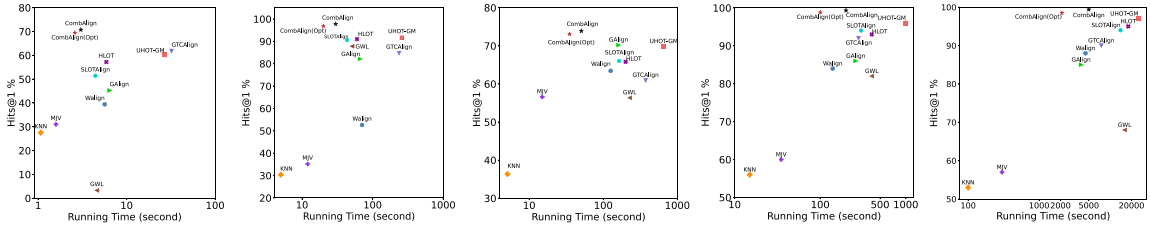
We show the results of the ablation study in Fig. 5. It is clear that all proposed modules have non-negligible contributions to model accuracy. More specifically, the Combine module boosts the accuracy on Douban (resp. ACM-DBLP) by more than two percentage points. The non-uniform marginal and feature transformation significantly improve the model performance.

3) Further Evaluation of Different Modules: We conduct a more extensive investigation of the proposed modules.

GRAFT **with different GNNs**. According to the theoretical results of Corollary IV.2, more expressive GNNs should lead to better model accuracy. We employ the CombAlign w/o C+NUM variant to test the lightweight GCN, GCN, and GIN (Cf. Section IV-B1). We also include an efficient Graph Transformer [67] with even more expressive power. The results nicely matches our theoretical conclusion (see Table VI).

Note that by default the GRAFT module uses GCN (e.g., for Table IV). We also point out that the GIN variant in Table VI has already outperformed all baselines shown in Table IV. We believe there is still room for further improvement with more powerful GNNs (and Graph Transformers).

Baselines with the Combine *module:* Next, we validate our Combine module for improving existing OT-based solutions. We substitute the GRAFT module with GWL, SLOTAlign and UHOT-GM, and report the final prediction with Combine. As shown in Table VII, all methods achieve a significant enhancement of Hits@1, further validating the effectiveness of our Combine module.



(a) Douban (b) Allmovie-Imdb (c) ACM-DBLP (d) CS (e) Physics

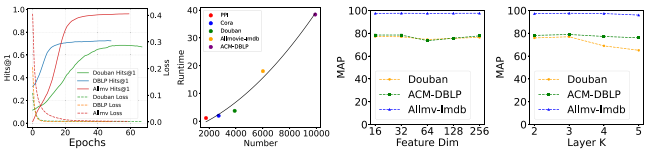
Fig. 6. Running time vs. Hits@1 of different methods.

TABLE VII
IMPROVING OTHER OT-BASED MODELS W/ `Combine` (HITS@1)

Methods	Douban	ACM-DBLP	Allmovie-Imdb
GWL	3.29	56.36	87.82
GWL+C	5.81	63.46	90.07
SLOTAlign	51.43	66.04	90.60
SLOTAlign+C	60.01	69.20	91.03
UHOT-GM	60.23	69.89	91.73
UHOT-GM+C	63.17	71.32	92.45

TABLE VIII
THE RATIO OF THE REMAINING SET OVER GROUND TRUTH AFTER ELIMINATING ONE-TO-MANY PREDICTIONS

CombAlign	Douban	ACM-DBLP	Allmovie-Imdb
- w/ C	**100%**	**100%**	**100%**
- w/o C	94.36%	88.36%	98.54%
- w/o C+NUM	94.18%	86.50%	97.60%
- w/o C+NUM+FT	91.32%	83.27%	96.25%



(a) (b) (c) (d)

Fig. 7. Further Evaluation. (a) Convergence. (b) Runtime of `CombAlign (Opt)` vs. graph size. (c) and (d) represent the sensitivity analysis.

One-to-one matching property of `CombAlign`*:* In Table VIII, we show the ratio of the remaining set compared to the ground truth set after eliminating one-to-many predictions for all model variants of `CombAlign`. This step is similar to that of Fig. 4 (see Section IV-C), in which we demonstrate the ratio of one-to-many predictions for our baselines. Consistent with our analysis, `CombAlign` manages to return a set of node pairs with one-to-one matching. Other variants of `CombAlign` also ensures higher similarity between matched pairs compared to the baselines.

4) Convergence Analysis: We simultaneously visualize the prediction accuracy and the model loss (i.e., Equation (1)) on three real-world datasets at different epochs, as shown in Fig. 7(a). With the increase of epochs, the loss gradually decreases to almost approaching zero, meanwhile, the Hits@1 value gradually converges.

5) Model Efficiency and Scalability: For efficiency analysis, we compare the running time of all methods in Fig. 6 on five datasets. It can be observed that `KNN` and `MJV` achieve the highest efficiency but suffer from poor performance. Our optimized algorithm, `CombAlign (Opt)`, has the best efficiency among all learning-based methods with better accuracy. Moreover, compared to the best-performing baselines, the speedup is about $10\times$. Even the `CombAlign` algorithm outperforms most baselines in terms of efficiency. This can be attributed to the strong expressive power of our model, which facilitates faster convergence. Note that the GW learning process takes the majority of the time, while the additional costs of `WL` and `Combine` modules are negligible. The fitted runtime curve versus graph size (Fig. 7(b)) aligns with the algorithm's quadratic time complexity.

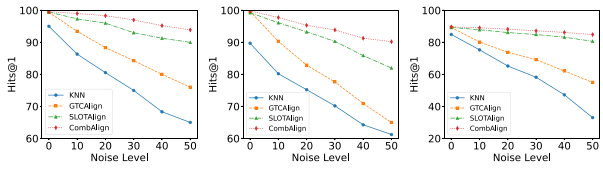
TABLE IX
COMPARISON OF HITS@1 ON THE DATASETS WITH 1 MILLION NODES

dataset	S3GA-RCF	CombAlign
EN-FR	71.77 (879 secs)	72.98 (903 secs)
EN-DE	71.93 (412 secs)	73.51 (427 secs)

To test the model's scalability on large-scale graphs, we integrate our solution with a divide-and-conquer framework, `S3GA-RCF` [68]. We first partition the source and target graphs into smaller subgraphs and apply our model to the subgraphs, and then aggregate the matching results. As shown in Table IX, EN-FR and EN-DE represent two datasets with one million nodes [68], where our model achieves higher accuracy with comparable time cost, validating its effectiveness. It is worth noting that the quality of the partitioning significantly affects the overall accuracy.

6) Sensitivity Analysis: To examine the impact of different hyper-parameters, on three real datasets including Douban, ACM-DBLP and Allmovie-Imdb, we analyze the performance of `CombAlign` with different feature dims $d \in \{16, 32, 64, 128, 256\}$ and the number of graph convolution layers $K \in \{2, 3, 4, 5\}$, as illustrated in Fig. 7(c) and (d). In general, the model performance is stable w.r.t. d. Meanwhile, the accuracy declines when K exceeds 3, consistent with the oversmoothing phenomenon of GNNs.

7) Robustness Analysis: To validate the model's robustness, we follow prior work [17], [19] and perturb varying proportions of edges on three synthetic datasets. As shown in Fig. 8, in which we select the representative method among traditional, embedding-based and OT-based solutions, when we increase the proportion of perturbed edges, the performance of all models



(a) Cora (b) Citeseer (c) PPI

Fig. 8. Noise vs. accuracy.

gradually decline. When the perturbation ratio is relatively low (e.g., below 10%), all methods, including KNN, achieve relatively high accuracy. Our model, CombAlign, demonstrates stronger robustness even with a high noise level. In this case, learning-based approaches demonstratre clear advantages over traditional algorithms such as KNN even on these synthetic datasets.

VI. Conclusion and Future Work

In this paper, we formally investigate the model expressiveness for unsupervised graph alignment from two theoretical perspectives. We characterize the model's discriminative power as correctly distinguishing matched and unmatched node pairs across graphs and study the model's capability of guaranteeing important matching properties including one-to-one node matching and mutual alignment. Then, we propose a hybrid model named CombAlign to combine the advantages of both optimal transport and embedding-based solutions, which leads to provably more discriminative power. Our method is further empowered by ensembling both OT and embedding-based predictions with a traditional algorithm-inspired strategy, i.e., maximum weight matching, to guarantee the matching properties. Extensive experiments demonstrate significant improvements in alignment accuracy and the effectiveness of the proposed modules. Moreover, our theoretical analysis of expressive power is confirmed. For future work, we aim to enhance the model's efficiency on large-scale graphs.

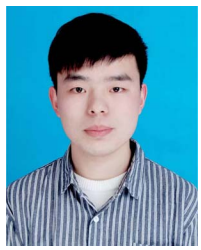
Acknowledgment

The authors would like to thank Prof. Lei Chen, Prof. Xiaofang Zhou, and Prof. Zhi Jin for fruitful discussions, and Haoran Cheng, Prof. Dixin Luo, and Prof. Hongteng Xu for their help in providing the baseline codes.

References

[1] J. Zhang and S. Y. Philip, "Multiple anonymized social networks alignment," in *Proc. IEEE Int. Conf. Data Mining*, 2015, pp. 599–608.
[2] C. Li et al., "Partially shared adversarial learning for semi-supervised multi-platform user identity linkage," in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, 2019, pp. 249–258.
[3] J. Tang et al., "Arnetminer: Extraction and mining of academic social networks," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2008, pp. 990–998.
[4] S. Zhang et al., "Balancing consistency and disparity in network alignment," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2021, pp. 2212–2222.
[5] H. Xu et al., "Gromov–Wasserstein learning for graph matching and node embedding," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6932–6941.
[6] R. Singh, J. Xu, and B. Berger, "Global alignment of multiple protein interaction networks with application to functional orthology detection," *Proc. Nat. Acad. Sci. USA*, vol. 105, no. 35, pp. 12763–12768, 2008.
[7] C. Piao et al., "Computing graph edit distance via neural graph matching," *Proc. VLDB Endowment*, vol. 16, no. 8, pp. 1817–1829, 2023.
[8] F. Bernard et al., "A solution for multi-alignment by transformation synchronisation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 2161–2169.
[9] S. Haller et al., "A comparative study of graph matching algorithms in computer vision," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 636–653.
[10] Z. Zhou et al., "Metabolite annotation from knowns to unknowns through knowledge-guided multi-layer metabolic networking," *Nature Commun.*, vol. 13, no. 1, 2022, Art. no. 6656.
[11] S. Feizi et al., "Spectral alignment of graphs," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 3, pp. 1182–1197, 3rd Quart., 2019.
[12] P. A. Karakasis et al., "Joint graph embedding and alignment with spectral pivot," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2021, pp. 851–859.
[13] J. Peng et al., "A new relaxation framework for quadratic assignment problems based on matrix splitting," *Math. Program. Computation*, vol. 2, pp. 59–77, 2010.
[14] H. T. Trung et al., "Adaptive network alignment with unsupervised and multi-order convolutional networks," in *Proc. IEEE 36th Int. Conf. Data Eng.*, 2020, pp. 85–96.
[15] J. Gao et al., "Unsupervised graph alignment with wasserstein distance discriminator," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2021, pp. 426–435.
[16] C. Wang et al., "GTCalign: Global topology consistency-based graph alignment," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 5, pp. 2009–2025, May 2024.
[17] J. Tang et al., "Robust attributed graph alignment via joint structure learning and optimal transport," in *Proc. IEEE 39th Int. Conf. Data Eng.*, 2023, pp. 1638–1651.
[18] V. Titouan et al., "Optimal transport for structured data with application on graphs," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6275–6284.
[19] H. Cheng et al., "DHOT-GM: Robust graph matching using a differentiable hierarchical optimal transport framework," 2023, *arXiv:2310.12081*.
[20] Y. Yan et al., "Hypergraph learning for unsupervised graph alignment via optimal transport," in *Proc. AAAI Conf. Artif. Intell.*, 2025, pp. 21913–21921.
[21] C. Villani et al., *Optimal Transport: Old and New*. Berlin, Germany: Springer, 2009.
[22] G. Peyré et al., "Gromov–Wasserstein averaging of kernel and distance matrices," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 2664–2672.
[23] M. Kolář et al., "Graphalignment: Bayesian pairwise alignment of biological networks," *BMC Syst. Biol.*, vol. 6, pp. 1–12, 2012.
[24] Z. Zeng et al., "Parrot: Position-aware regularized optimal transport for network alignment," in *Proc. ACM Web Conf.*, 2023, pp. 372–382.
[25] S. Zhang and H. Tong, "Final: Fast attributed network alignment," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2016, pp. 1345–1354.
[26] Y. Yan et al., "Bright: A bridging algorithm for network alignment," in *Proc. Web Conf.*, 2021, pp. 3907–3917.
[27] L. Liu et al., "WL-align: Weisfeiler–Lehman relabeling for aligning users across networks via regularized representation learning," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 1, pp. 445–458, Jan. 2024.
[28] N. Ning, B. Wu, H. Ren, and Q. Li, "Graph alignment neural network model with graph to sequence learning," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 9, pp. 4693–4706, Sep. 2024.
[29] T. T. Huynh et al., "Network alignment with holistic embeddings," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 2, pp. 1881–1894, Feb. 2023.
[30] N. Shervashidze et al., "Weisfeiler–Lehman graph kernels," *J. Mach. Learn. Res.*, vol. 12, no. 9, pp. 2539–2561, 2011.
[31] J. Edmonds et al., "Theoretical improvements in algorithmic efficiency for network flow problems," *J. ACM*, vol. 19, no. 2, pp. 248–264, 1972.
[32] R. Jonker and T. Volgenant, "A shortest augmenting path algorithm for dense and sparse linear assignment problems," in *Proc. 16th Annu. Meeting DGOR Cooperation NSOR/Vorträge Der*, 1988, pp. 622–622.
[33] Z.-H. Zhou, *Ensemble methods: Foundations and algorithms*. Boca Raton, FL, USA: CRC Press, 2012.
[34] T. G. Dietterich, "Ensemble methods in machine learning," in *Proc. Int. Workshop Mult. Classifier Syst.*, 2000, pp. 1–15.
[35] H. Nassar et al., "Low rank spectral network alignment," in *Proc. 2018 World Wide Web Conf.*, 2018, pp. 619–628.

[36] J. Hermanns et al., "Grasp: Graph alignment through spectral signatures," in *Proc. 5th Int. Joint Conf. Web Big Data*, Guangzhou, China, 2021, pp. 44–52.
[37] L. Liu et al., "Aligning users across social networks using network embedding," in *Proc. Int. Joint Conf. Artif. Intell.*, 2016, pp. 1774–80.
[38] Q. Sun et al., "Towards higher-order topological consistency for unsupervised network alignment," in *Proc. IEEE 39th Int. Conf. Data Eng.*, 2023, pp. 177–190.
[39] J. Peng et al., "Robust network alignment with the combination of structure and attribute embeddings," in *Proc. 2023 IEEE Int. Conf. Data Mining*, 2023, pp. 498–507.
[40] D. Koutra et al., "Big-align: Fast bipartite graph alignment," in *Proc. IEEE 13th Int. Conf. Data Mining*, 2013, pp. 389–398.
[41] S. Zhang et al., "Nettrans: Neural cross-network transformation," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2020, pp. 986–996.
[42] M. Heimann et al., "Regal: Representation learning-based graph alignment," in *Proc. 27th ACM Int. Conf. Inf. Knowl. Manage.*, 2018, pp. 117–126.
[43] Q. Zhou et al., "Attent: Active attributed network alignment," in *Proc. Web Conf.*, 2021, pp. 3896–3906.
[44] H. Hong et al., "Domain-adversarial network alignment," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 7, pp. 3211–3224, Jul. 2022.
[45] S. Chen et al., "Enhancing robust semi-supervised graph alignment via adaptive optimal transport," *World Wide Web*, vol. 28, no. 2, 2025, Art. no. 22.
[46] Z. Wang et al., "Cross-lingual knowledge graph alignment via graph convolutional networks," in *Proc. 2018 Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 349–357.
[47] Z. Sun et al., "Bootstrapping entity alignment with knowledge graph embedding," in *Proc. Int. Joint Conf. Artif. Intell.*, vol. 18, 2018.
[48] K. Zeng et al., "A comprehensive survey of entity alignment for knowledge graphs," *AI Open*, vol. 2, pp. 1–13, 2021.
[49] R. Zhang et al., "A benchmark and comprehensive survey on knowledge graph entity alignment via representation learning," *VLDB J.*, vol. 31, no. 5, pp. 1143–1168, 2022.
[50] J. Tang et al., "A fused gromov-wasserstein framework for unsupervised knowledge graph entity alignment," 2023, *arXiv:2305.06574*.
[51] X. Jiang et al., "Unlocking the power of large language models for entity alignment," 2024, *arXiv:2402.15048*.
[52] R. Zhao et al., "Towards unsupervised entity alignment for highly heterogeneous knowledge graphs," in *Proc. IEEE 41st Int. Conf. Data Eng.*, 2025, pp. 3792–3806.
[53] S. Chen et al., "Entity alignment with noisy annotations from large language models," in *Proc. Adv. Neural Inf. Process. Syst.*, 2024, pp. 15097–15120.
[54] M. Cuturi, "Sinkhorn distances: Lightspeed computation of optimal transport," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 2292–2300.
[55] R. Sinkhorn et al., "Concerning nonnegative matrices and doubly stochastic matrices," *Pacific J. Math.*, vol. 21, no. 2, pp. 343–348, 1967.
[56] Y. Xie et al., "A fast proximal point method for computing exact wasserstein distance," in *Proc. Conf. Uncertainty Artif. Intell.*, 2020, pp. 433–453.
[57] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*.
[58] P. Veličković et al., "Graph attention networks," 2017, *arXiv:1710.10903*.
[59] K. Xu et al., "How powerful are graph neural networks?," 2018, *arXiv:1810.00826*.
[60] F. Wu et al., "Simplifying graph convolutional networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6861–6871.
[61] D. F. Crouse et al., "On implementing 2D rectangular assignment algorithms," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 52, no. 4, pp. 1679–1696, Aug. 2016.
[62] J. Munkres, "Algorithms for the assignment and transportation problems," *J. Soc. Ind. Appl. Math.*, vol. 5, no. 1, pp. 32–38, 1957.
[63] H. Cui et al., "Solving large-scale assignment problems by kuhn-munkres algorithm," in *Proc. 2nd Int. Conf. Adv. Mech. Eng. Ind. Informat.*, 2016, pp. 822–827.
[64] B. Schwartz, "A computational analysis of the auction algorithm," *Eur. J. Oper. Res.*, vol. 74, no. 1, pp. 161–169, 1994.
[65] L. Buš and P. Tvrdık, "Towards auction algorithms for large dense assignment problems," *Comput. Optim. Appl.*, vol. 43, pp. 411–436, 2009.
[66] S. Wang et al., "Fora: Simple and effective approximate single-source personalized pagerank," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2017, pp. 505–514.
[67] Q. Wu et al., "SGFormer: Simplifying and empowering transformers for large-graph representations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2023, pp. 64753–64773.
[68] W. Guo et al., "S3GA: Towards scalable self-supervised learning for large scale graph alignment," in *Proc. 31st ACM SIGKDD Conf. Knowl. Discov. Data Mining V. 2*, 2025, pp. 755–766.
[69] S. Zhang and H. Tong, "Attributed network alignment: Problem definitions and fast solutions," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 9, pp. 1680–1692, Sep. 2018.
[70] P. Sen et al., "Collective classification in network data," *AI Mag.*, vol. 29, no. 3, pp. 93–93, 2008.
[71] M. Zitnik et al., "Predicting multicellular function through multi-layer tissue networks," *Bioinformatics*, vol. 33, no. 14, pp. i190–i198, 2017.
[72] O. Shchur et al., "Pitfalls of graph neural network evaluation," 2018, *arXiv:1811.05868*.
[73] I. Gulrajani et al., "Improved training of wasserstein GANs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5769–5779.



Songyang Chen received the BS degree in automation from Yangtze University, in 2018. He is currently working toward the PhD degree with the School of Computer Science and Technology, Beijing Jiaotong University. His research interests include graph data mining, with a particular focus on graph alignment, entity alignment, and subgraph matching.

Yu Liu received the BS degree in computer science from Shandong University, in 2011, and the MEng and PhD degrees from the School of Information, Renmin University, in 2014 and 2018, respectively. He is now an associate professor with Beijing Jiaotong University. He has published more than 20 refereed papers in various journals and conferences. His current research interests include scalable graph algorithms and graph learning methods.

Lei Zou received the BS and PhD degrees in computer science from the Huazhong University of Science and Technology (HUST), in 2003 and 2009, respectively. Now, he is a professor with the Wangxuan Institute of Computer Technology, Peking University. His research interests include graph database and semantic data management.

Zexuan Wang received the BE degree in computer science and technology from Beijing Jiaotong University, in 2023. He is currently working toward the MS degree with the School of Computer Science and Technology, Beijing Jiaotong University. His research interests include graph computing, with a special emphasis on graph alignment, and subgraph isomorphism.

Youfang Lin received the PhD degree in signal and information processing from Beijing Jiaotong University, Beijing, China, in 2003. He is a professor with the School of Computer Science and Technology, Beijing Jiaotong University. His main fields of expertise and current research interests include Big Data technology, intelligent systems, complex networks, and traffic data mining.