On the Adversarial Robustness of Machine Learning Models on Multi-Graph Scenarios

by

Zijie Zhang

A dissertation submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Auburn, Alabama
August 5, 2023

Keywords: Graph, Adversarial, Deep Learning

Abstract

In this work, we study the the task of graph matching under several scenarios in an adversarial context. Despite achieving remarkable performance, deep learning based graph matching still suffers from harassment caused by small adversarial perturbations. This perturbation, usually delivered in the form of small yet elaborately designed alternations of the topology of the target graphs, i.e., addition and deletion of very few portion of the edges, can result in serious degradation of the performance of the graph matching process. To begin our investigation with, we designed a density based and meta-learning enhanced attack specifically for graph matching and observed high mismatching rate in empirical analysis. In addition, we also showed that graph models adversarial trained on the attacking perturbation generated using the above approach also gained extra robustness. The weakness of this method as a defense against the adversarial attacks is that it does not provide any kind of guarantee in the sense of unaffected behavior under attacks with limited perturbation budget. Thus, we went further with Lipschitz networks equipped with specially designed Kl-Lipschitz Weibull activation combined with weights constrained calculated target norms with polar decomposition techniques to provide provable robustness while persevering the expressiveness of the network to mitigate the inevitable loss of matching rate. We also investigated the potential threats when performing graph matching in a Federated Learning scheme. An algorithm is proposed to deals with the dilemma in this specific problem which being the data privacy constraint requiring graphs not being shared with each other on one side and the nature of the graph matching problem demanding the possession of the knowledge of at least two graph simultaneously.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problem Overview

1.1.1 Graph Matching

Graph matching is one of the most important research topics in the graph domain, which aims to match the same entities (i.e., nodes) across two or more graphs [202, 207, 162, 165, 167, 188, 172, 213, 15, 191]. It has been widely applied to many real-world applications ranging from protein network matching in bioinformatics [154, 180], user account linking in different social networks [179, 169, 209, 157, 105, 144, 158], and knowledge translation in multilingual knowledge bases [199, 214], to geometric keypoint matching in computer vision [27]. Existing research efforts on graph matching can be classified into three broad categories: (1) structure-based techniques, which rely only upon the topological information to match two or multiple input graphs [162, 59, 205, 165, 172, 213, 15, 100, 87, 159, 183, 175, 158, 146]; (2) attribute-based approaches, which utilize highly discriminative structure and/or attribute features for ensuring the matching effectiveness [99, 204, 169, 140, 181, 142, 91, 209, 35, 49, 21, 206, 168, 170, 27]; and (3) heterogeneous methods, which employ heterogeneous structural, content, spatial, and temporal features to further improve the matching performance [203, 155, 163, 207, 197, 208, 193, 176, 210, 211, 144].

Formally, graph matching is usually defined as follows. Given graphs $\{G_1, G_2, G_3, ..., G_i\}$, each consists of a node set V_i and a edge set E_i, the algorithm is expected to produce an injective mapping $M_{s \to t}$ given any two graphs G_s and G_t, where $s \neq t$ and $s, t \in \{1...i\}$, such that the similarity between a node v in G_s and its predicted counterpart $M_{s \to t}(v)$ in G_t is maximized. The definition of this similarity can vary according to the details of the scenario. In supervised settings, the similarity is most often designated as the hit rate of the predicted matches of the nodes pairs in the provided ground truth. In unsupervised settings, various kinds of topological similarity (e.g. the consistency of the neighborhood of the two matching nodes) are usually adopted.

1.1.2 Adversarial Perturbations

Despite the skyrocketing capabilities of neural networks in recent years, many have observed that they are quite sensitive to little changes in the input data at the same time. This flaw can be maliciously exploited to lead a well-trained neural network to produce unexpected results. [314] showed that the classification result of a image can be easily manipulated via the addition of a small amount of noise to the original pixels, which is essentially imperceptible to human eyes. Later, works like [315] and [316] achieved similar results on text and audio datasets. It is clear that adversarial perturbations are posing a serious threat on the reliability and trustworthiness of deep learning based methods as a whole.

1.1.3 Graph Matching in an Adversarial Context

Recent literature has shown that both traditional and deep graph learning algorithms remain highly sensitive to adversarial attacks, i.e., carefully designed small perturbations in graph structure and attributes can cause the models to produce wrong prediction results [18, 216, 182, 215, 132, 185, 201, 190, 164, 198, 196, 217]. We have witnessed various effective attack models to cause failures of node classification [18, 216, 190, 88, 215, 187, 23, 186], community detection [139, 194, 137, 160], network embedding [136, 9, 135], link prediction [212], similarity search [141], malware detection [151], and knowledge graph embedding [201]. However, there is still a paucity of analyses of the vulnerability of graph matching under adversarial attacks, which is much more difficult to study. Most of the existing models to fool other graph learning tasks conduct the adversarial attacks on a single graph but the graph matching task analyzes both intra-graph and inter-graph interactions of multiple graphs.

1.2 Road Maps

1.2.1 Adversarial Attacks and Training

In our first work, we specifically investigate the problem within the context of graph matching. we proposed an adversarial attack model with two novel attack techniques to perturb the graph structure and degrade the quality of deep graph matching: (1) a kernel density estimation approach is utilized to estimate and maximize node densities to derive imperceptible perturbations, by pushing attacked nodes to dense regions in two graphs, such that they are indistinguishable from many neighbors; and (2) a meta learning-based projected gradient descent method is developed to well choose attack starting points and to improve the search performance for producing effective perturbations. We evaluate the effectiveness of the attack model on real datasets and validate that the attacks can be transferable to other graph learning models.

1.2.2 Certifiable Robustness

In our second work, we tried to construct a scheme of defense which is capable of tackling a broader scope of attacks. An attack agnostic graph-adaptive 1-Lipschitz neural network which we named ERNN is proposed for improving the robustness of deep multiple graph learning while achieving remarkable expressive power. A Kl-Lipschitz Weibull activation function $\bar{f}$ is designed to enforce the gradient norm $\|\nabla \bar{f}(x)\|$ as K_l at layer l. The nearest matrix orthogonalization and polar decomposition techniques are utilized to constraint the weight norm $\|\bar{W_l}\|$ as $1/K_l$ and make $\bar{W_l}$ close to the original weight W_l . The theoretical analysis is conducted to derive lower and upper bounds of feasible K_l under the 1-Lipschitz constraint. The combination of norm-constrained $\bar{f}$ and $\bar{W_l}$ leads to the 1-Lipschitz neural network for expressive and robust multiple graph learning.

1.2.3 Defending Graph Matching in Federated Learning

Federated graph learning, which aims to solve graph learning problem on distributed graph data, has led to state-of-the-art innovations. Graph matching in the setting of federated learning is still an open problem. In our last work, we propose an unsupervised federated graph matching algorithm, UFGM, for inferring matched node pairs on different graphs across clients while maintaining privacy requirement, by leveraging graphlet theory and trust region optimization. First, the nodes' graphlet features are captured to generate pseudo matched node pairs on different graphs across clients as pseudo training data for tackling the dilemma of unsupervised graph matching in federated setting and leveraging the strength of supervised graph matching. An approximate graphlet enumeration method is proposed to sample a small number of graphlets and capture nodes' graphlet features. Theoretical analysis is conducted to demonstrate that the approximate method is able to maintain the quality of graphlet estimation while reducing its expensive cost. Second, we propose a separate trust region algorithm for pseudo supervised federated graph matching while maintaining the privacy constraints. In order to avoid expensive cost of the second-order Hessian computation in the trust region algorithm, we propose two weak quasi-Newton conditions to construct a positive definite scalar matrix as the Hessian approximation with only first-order gradients. We theoretically derive the error introduced by the separate trust region due to the Hessian approximation and conduct the convergence analysis of the approximation method.

Chapter 2

Adversarial Attacks on Deep Graph Matching

2.1 Introduction

In this work, we aim to answer the following questions: (1) Are graph matching algorithms sensitive to small perturbation of graph structure? (2) How do we develop effective and imperceptible perturbations for degrading the performance of deep graph matching models?

A large number of research advances in adversarial attacks on graph data utilize iterative gradient-based methods to produce effective adversarial perturbations that fool a graph learning model [184, 192, 18, 215, 88, 187, 200, 138]. However, a recent study reports that the iterative gradient-based methods, such as Fast Gradient Sign Method (FGSM) [148] and Projected Gradient Descent (PGD) [166], start the attacks from original examples and add perturbations monotonically along the direction of gradient descent, resulting in a lack of diversity and adaptability of generated iterative trajectories [178]. This often leads to invalid attacks since the iterative trajectories have difficulties crossing decision boundary of target learning model with small perturbation. Can we find a shortcut across the decision boundary to derive more effective attacks by beginning from good attack starting points in the graph matching?

Traditionally, graph matching techniques are based on the assumption of feature consistency across graphs: Two nodes in different graphs are more likely to be found to be matching if they have similar topological and/or attribute features in respective graphs [207, 99, 140, 143, 35, 100]. These methods compute the similarity (or distance) scores between pairwise nodes across graphs and choose the node pairs with largest similarity (or smallest distance) as matching results [105, 91, 49, 158]. Intuitively, if an attacker perturbs a node by throwing it into a dense region in the graph with many similar nodes, i.e., a pile of nodes similar to each other, such that this attacked node is similar to many neighbors, then it is hard for humans or defender programs to recognize it from the node pile. In addition, if two matched nodes are simultaneously moved to such dense regions in respective graphs, then this dramatically increases the difficulty in matching them correctly among many similar candidate nodes.

To our best knowledge, this work is the first to study adversarial attacks on graph matching.

We propose to utilize kernel density estimation (KDE) technique to estimate the probability density function of nodes in two graphs, to understand the intrinsic distribution of graphs. By maximizing the estimated densities of nodes to be attacked, we push them to dense regions in respective graphs to generate adversarial nodes that are indistinguishable from many neighbors in dense regions. This increases the chance of producing wrong matching results as well as reduces the risk of perturbations being detected by humans or by defender programs. Our analysis is the first to introduce the KDE technique to conduct imperceptible attacks on graph data.

Searching for good attack starting points in large graphs is computationally inefficient. We develop a meta learning-based projected gradient descent (MLPGD) model to quickly adapt to a variety of new search tasks on multiple batches of target nodes for deriving effective attacks. However, the MLPGD model is non-smooth and non-differential, as the perturbation is a multi-step process and the projection at each step is non-differential. A Gaussian smoothing method is designed to approximate a smoothed model, and a Monte Carlo REINFORCE method is used to estimate the model gradient.

Empirical evaluation on real datasets demonstrates the superior performance of the GMA model against several state-of-the-art adversarial attack methods on graph data. Moreover, we validate that the attack strategies are transferable to other popular graph learning models in Appendix **??**.

2.2 Problem Definition

Given two graphs G^1 and G^2 to be matched, each is denoted as $G^s = (V^s, E^s)$ ($s = 1$ or 2), where $V^s = \{v_1^s, \cdots, v_{N^s}^s\}$ is the set of N^s nodes and $E^s = \{(v_i^s, v_j^s) : 1 \leq i, j \leq N^s\}$ is the set of edges. Each G^s has an $N^s \times N^s$ binary adjacency matrix $\mathbf{A}^s$, where each entry $\mathbf{A}_{ij}^s = 1$ if there exists an edge $(v_i^s, v_j^s) \in E^s$; otherwise $\mathbf{A}_{ij}^s = 0$. $\mathbf{A}_{i:}^s$ specifies the i^{th} row vector of $\mathbf{A}^s$. In this paper, if there are no specific descriptions, we use $\mathbf{v}_i^s$ to denote a node v_i^s itself and its representation $\mathbf{A}_{i:}^s$, i.e., $\mathbf{v}_i^s = \mathbf{A}_{i:}^s$ and we utilize $\mathbf{v}_{ij}^s$ to specify the j^{th} dimension of $\mathbf{v}_i^s$, i.e., $\mathbf{v}_{ij}^s = \mathbf{A}_{ij}^s$.

The dataset is divided into two disjoint sets D' and D. The former denotes a set of known matched node pairs $D' = \{(\mathbf{v}_i^1, \mathbf{v}_k^2) | \mathbf{v}_i^1 \leftrightarrow \mathbf{v}_k^2, \mathbf{v}_i^1 \in V^1, \mathbf{v}_k^2 \in V^2\}$, where $\mathbf{v}_i^1 \leftrightarrow \mathbf{v}_k^2$ indicates that two nodes $\mathbf{v}_i^1$ and $\mathbf{v}_k^2$ belong to the same entity. The latter, denoted by $D = \{(\mathbf{v}_i^1, \mathbf{v}_k^2) | \mathbf{v}_i^1 \leftrightarrow \mathbf{v}_k^2, \mathbf{v}_i^1 \in V^1, \mathbf{v}_k^2 \in V^2\}$, is used to evaluate the graph matching performance, where the nodes (but not their matchings) are also observed during training. The goal of graph matching is to utilize D' as the training data to identify the one-to-one matching relationships between nodes $\mathbf{v}_i^1$ and $\mathbf{v}_k^2$ in the test data D. By following the same idea in existing efforts [105, 91, 49, 158], this paper aims to minimize the distances between projected source nodes $M(\mathbf{v}_i^1) \in D'$ and target ones $\mathbf{v}_k^2 \in D'$. The node pairs $(\mathbf{v}_i^1, \mathbf{v}_k^2) \in D$ with the smallest distances are selected as the matching results.

$$\min_M L \quad \text{where} \quad L = \mathbb{E}_{(\mathbf{v}_i^1, \mathbf{v}_k^2) \in D'} \| M(\mathbf{v}_i^1) - \mathbf{v}_k^2 \|_2^2 \tag{2.1}$$

where M denotes an injective one-to-one matching function $M : \mathbf{v}_i^1 \in V^1 \mapsto \mathbf{v}_k^2 \in V^2$.

The adversarial attack problem is defined as maximally degrading the matching performance of M on the test data D by injecting edge perturbations (including edge insertion and deletion) into $G^s = (V^s, E^s)$ ($s = 1$ or 2), leading to two adversarial graphs $\hat{G}^s = (\hat{V}^s, \hat{E}^s)$. We assume the attacker has limited capability, so that he/she can only make small perturbations.

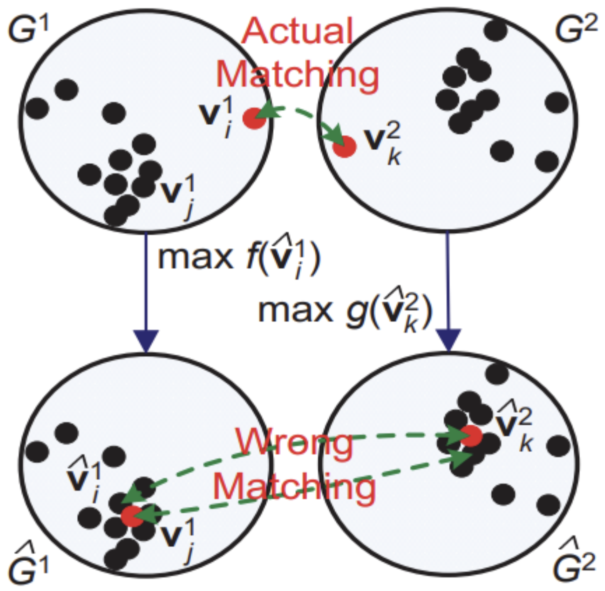


Figure 2.1: Imperceptible Attacks

2.3 Imperceptible Attacks with Node Density Estimation and Maximization

Intuitively, in Eq.(2.1), if there exist nodes $\mathbf{v}_j^1$ similar to $\mathbf{v}_i^1$, i.e., $\mathbf{v}_j^1 \approx \mathbf{v}_i^1$, such that $\|M(\mathbf{v}_j^1) - \mathbf{v}_k^2\|_2^2 < \|M(\mathbf{v}_i^1) - \mathbf{v}_k^2\|_2^2$, then a wrong matching $(\mathbf{v}_j^1, \mathbf{v}_k^2)$ will be generated. In addition, if there are many such $\mathbf{v}_j^1$s around $\mathbf{v}_i^1$, then it is hard to recognize $\mathbf{v}_i^1$ from a pile of similar nodes. Thus, if we move $\mathbf{v}_i^1$ to dense regions that contain many similar $\mathbf{v}_j^1$s, then this dramatically increases the possibility of deriving the wrong matching $(\mathbf{v}_j^1, \mathbf{v}_k^2)$ among many similar candidate nodes. Also, as many $\mathbf{v}_j^1$s are around the adversarial node $\hat{\mathbf{v}}_i^1$, it is difficult for humans or defender programs to detect $\hat{\mathbf{v}}_i^1$, as shown in a toy example in Figure 2.1.

Motivated by this, we propose to employ kernel density estimation (KDE) method to generate imperceptible perturbations. In statistics, the KDE is to estimate the probability density function $f(x)$ of a random variable x with unknown distribution [173]. It helps reveal the intrinsic distribution.

Concretely, let $\mathbf{v}^1$ be a N^1-dimensional random variable to denote all nodes $\{\mathbf{v}_i^1, \cdots, \mathbf{v}_{N^1}^1\}$ in graph G^1 with an unknown density f. A function $\hat{f}(x)$ is estimated to best approximate $f(x)$.

$$\hat{f}(\mathbf{v}^1) = \frac{1}{N^1 \det(\mathbf{B})} \sum_{i=1}^{N^1} \mathcal{K}\left(\mathbf{B}^{-1}\left(\mathbf{v}^1 - \mathbf{v}_i^1\right)\right) \tag{2.2}$$

where $\det(\cdot)$ denotes the determinant operation. $\mathbf{B} > 0$ is a bandwidth to be estimated. It is an $N^1 \times N^1$ diagonal matrix $\mathbf{B} = diag(b_1, \cdots, b_{N^1})$, which has strong influence on the density estimation $\hat{f}(\mathbf{v}^1)$. A good $\mathbf{B}$ should be as small as the data can allow. $\mathcal{K}$ is a product symmetric kernel that satisfies $\int \mathcal{K}(x)dx = 1$ and $\int x\mathcal{K}(x)dx = 0$. The above vector-wise form $\hat{f}(\mathbf{v}^1)$ can be rewritten as an element-wise form, where $\mathbf{v}_j^1$ represents the j^{th} dimension in $\mathbf{v}^1$.

$$\hat{f}(\mathbf{v}^1) = \frac{1}{N^1} \sum_{i=1}^{N^1} \prod_{j=1}^{N^1} \frac{1}{b_j} \mathcal{K}\Big(\frac{\mathbf{v}_j^1 - \mathbf{v}_{ij}^1}{b_j}\Big) \tag{2.3}$$

The derivative $\frac{\partial \hat{f}(\mathbf{v}^1)}{\partial b_j}$ w.r.t. each bandwidth b_j in $\mathbf{B}$ is computed as follows, where $K(x) = \frac{d \log \mathcal{K}(x)}{dx}$.

$$\frac{\partial \hat{f}(\mathbf{v}^1)}{\partial b_j} = \frac{1}{N^1} \sum_{i=1}^{N^1} \frac{\partial \Big[\prod_{l=1}^{N^1} \frac{1}{b_l} \mathcal{K}\big(\frac{\mathbf{v}_l^1 - \mathbf{v}_{il}^1}{b_l}\big)\Big]}{\partial b_j} = -\frac{1}{N^1} \sum_{i=1}^{N^1} \Big(\frac{1}{b_j} + \frac{\mathbf{v}_l^1 - \mathbf{v}_{il}^1}{b_j^2} K\big(\frac{\mathbf{v}_l^1 - \mathbf{v}_{il}^1}{b_j}\big)\Big) \prod_{l=1}^{N^1} \frac{1}{b_l} \mathcal{K}\big(\frac{\mathbf{v}_l^1 - \mathbf{v}_{il}^1}{b_l}\big) \tag{2.4}$$

Traditional KDE methods often fail on high-dimensional data [150, 177, 153, 156], when bandwidths need to be selected for each dimension. A greedy search method is utilized to select bandwidths in the KDE: If a dimension j is insignificant, then changing the bandwidth b_j for that dimension should have a weak impact on $\hat{f}(\mathbf{v}^1)$, while the changing b_j for an important j should cause a large change in $\hat{f}(\mathbf{v}^1)$. Fortunately, $\frac{\partial \hat{f}(\mathbf{v}^1)}{\partial b_j}$ can differentiate these two types of dimensions. Based on the above analysis, we greedily decrease b_j with a sequence $b_0, b_0 s, b_0 s^2, \cdots$ for a parameter $0 < s < 1$, until b_j is smaller than a certain threshold τ_j, to see if a small change in b_j can result in a large change in $\hat{f}(\mathbf{v}^1)$. The method also offers a good way to estimate $\big[\frac{\partial \hat{f}(\mathbf{v}^1)}{\partial b_1}, \cdots, \frac{\partial \hat{f}(\mathbf{v}^1)}{\partial b_{N^1}}\big]$ along a sparse path.

Concretely, $\hat{f}(\mathbf{v}^1)$ is estimated by beginning with an initial $\mathbf{B} = diag(b_0, \cdots, b_0)$ for a large b_0, and then estimate $\frac{\partial \hat{f}(\mathbf{v}^1)}{\partial b_j}$ as follows and decrease b_j if $\frac{\partial \hat{f}(\mathbf{v}^1)}{\partial b_j}$ is large.

$$\frac{\partial \hat{f}(\mathbf{v}^1)}{\partial b_j} = \frac{1}{N^1}\sum_{i=1}^{N^1} \frac{\partial \Big[\prod_{l=1}^{N^1} \frac{1}{b_l} \mathcal{K}\big(\frac{\mathbf{v}_l^1 - \mathbf{v}_{il}^1}{b_l}\big)\Big]}{\partial b_j} = \frac{1}{N^1}\sum_{i=1}^{N^1} \frac{\mathcal{K}\big(\frac{\mathbf{v}_j^1 - \mathbf{v}_{ij}^1}{b_j}\big)}{\mathcal{K}\big(\frac{\mathbf{v}_j^1 - \mathbf{v}_{ij}^1}{b_j}\big)} \prod_{l=1}^{N^1} \mathcal{K}\big(\frac{\mathbf{v}_l^1 - \mathbf{v}_{il}^1}{b_l}\big) = \frac{1}{N^1}\sum_{i=1}^{N^1} \frac{\partial \hat{f}(\mathbf{v}_i^1)}{\partial b_j} \tag{2.5}$$

The corresponding variance $\mathrm{Var}\Big(\frac{\partial \hat{f}(\mathbf{v}^1)}{\partial b_j}\Big)$ is given below.

$$\mathrm{Var}\Big(\frac{\partial \hat{f}(\mathbf{v}^1)}{\partial b_j}\Big) = \mathrm{Var}\Big(\frac{1}{N^1}\sum_{i=1}^{N^1} \frac{\partial \hat{f}(\mathbf{v}_i^1)}{\partial b_j}\Big) \tag{2.6}$$

In this work, assuming that the graph data follow the Gaussian distribution, a product Gaussian kernel $\mathcal{K}$ is used to estimate the node density $\hat{f}(\mathbf{v}^1)$. $\frac{\partial \hat{f}(\mathbf{v}^{\mathbf{1}})}{\partial b_j}$ is accordingly updated as follows.

$$\begin{aligned}\frac{\partial \hat{f}(\mathbf{v}^{\mathbf{1}})}{\partial b_j} &= \frac{C}{N^1}\sum_{i=1}^{N^1} \Big(\big(\mathbf{v}_j^1 - \mathbf{v}_{ij}^1\big)^2 - b_j^2 \Big) \prod_{l=1}^{N^1} \mathcal{K}\big(\frac{\mathbf{v}_l^1 - \mathbf{v}_{il}^1}{b_l}\big) \propto \frac{1}{N_1}\sum_{i=1}^{N_1} \Big(\big(\mathbf{v}_j^1 - \mathbf{v}_{ij}^1\big)^2 - b_j^2 \Big) \prod_{l=1}^{N^1} \mathcal{K}\big(\frac{\mathbf{v}_l^1 - \mathbf{v}_{il}^1}{b_l}\big) \\ &= \frac{1}{N^1}\sum_{i=1}^{N^1} \Big(\big(\mathbf{v}_j^1 - \mathbf{v}_{ij}^1\big)^2 - b_j^2 \Big) \exp\big(- \sum_{l=1}^{N^1} \frac{\big(\mathbf{v}_l^1 - \mathbf{v}_{il}^1\big)^2}{2b_j^2} \big)\end{aligned} \tag{2.7}$$

where C denotes a proportionality constant $C = \frac{1}{b_j^3} \prod_{l=1}^{N^1} \frac{1}{b_l}$. It can be safely ignored to avoid computation overflow when $b_l \to 0$ for large N^1. The bandwidth estimation is presented in Algorithm 1.

Based on the estimated $\mathbf{B}$ and the Gaussian kernel $\mathcal{K}$, the closed form of $\hat{f}(\mathbf{v}^{\mathbf{1}})$ is derived below.

$$\hat{f}(\mathbf{v^1}) = \frac{1}{N^1}\sum_{i=1}^{N^1}\prod_{j=1}^{N^1}\mathcal{K}\big(\frac{\mathbf{v}_j^1 - \mathbf{v}_{ij}^1}{b_j}\big)\sqrt{\frac{|\mathbf{B}+\mathbf{\Sigma}|}{|\mathbf{\Sigma}|}}\times\exp\big(-\frac{(\mathbf{v}^1-\mu)^T\,(\mathbf{\Sigma}^{-1} - (\mathbf{B}+\mathbf{\Sigma})^{-1})\,(\mathbf{v}^1-\mu)}{2}\big) \tag{2.8}$$

where μ and $\mathbf{\Sigma}$ are the maximum likelihood estimation of the mean vector and covariance matrix of the Gaussian distribution. Please refer to Appendices **??** and **??** for detailed derivation of $\hat{f}(\mathbf{v^1})$.

As two graphs G^1 and G^2 often have different structures and distributions and thus the same KDE method as Algorithm 1 is utilized to estimate the density $\hat{g}(\mathbf{v^2})$ of $\mathbf{v^2}$. Based on the estimations $\hat{f}(\mathbf{v^1})$ and $\hat{g}(\mathbf{v^2})$, the attacker aims to maximize the following loss $\mathcal{L}_D$ with imperceptible perturbations.

$$\mathcal{L}_D = \sum_{(\hat{\mathbf{v}}_i^1,\hat{\mathbf{v}}_k^2)\in D}\mathcal{L}(\hat{\mathbf{v}}_i^1,\hat{\mathbf{v}}_k^2) \;\text{ where }\; \mathcal{L}(\hat{\mathbf{v}}_i^1,\hat{\mathbf{v}}_k^2) = \|M(\hat{\mathbf{v}}_i^1) - \hat{\mathbf{v}}_k^2)\|_2^2 + \hat{f}(\hat{\mathbf{v}}_i^1) + \hat{g}(\hat{\mathbf{v}}_k^2) \tag{2.9}$$

where $\hat{\mathbf{v}}_i^1 = \mathbf{v}_i^1 + \delta_i^1$ (and $\hat{\mathbf{v}}_k^2 = \mathbf{v}_k^2 + \delta_k^2$) denote adversarial versions of clean nodes $\mathbf{v}_i^1$ (and $\mathbf{v}_k^2$) in G^1 (and G^2) by adding a small amount of edge perturbations δ_i^1 (and δ_k^2) through our proposed MLPGD method in the next section, such that $M(\hat{\mathbf{v}}_i^1)$ is far away from $\hat{\mathbf{v}}_k^2$ and thus the matching accuracy is decreased. In addition, we push $\mathbf{v}_i^1$ and $\mathbf{v}_k^2$ to dense regions to generate $\hat{\mathbf{v}}_i^1$ and $\hat{\mathbf{v}}_k^2$, by maximizing $\hat{f}(\hat{\mathbf{v}}_i^1)$ and $\hat{g}(\hat{\mathbf{v}}_k^2)$, such that $\hat{\mathbf{v}}_i^1$ and $\hat{\mathbf{v}}_k^2$ are indistinguishable from their neighbors in perturbed graphs. This reduces the possibility of perturbation detection by humans or defender programs.

2.4 Effective Attacks via Meta Learning-based Projected Gradient Descent

In Figure 2.2, two dashed purple curves denote the decision boundary of graph matching. If we move a clean node $\mathbf{v}_i^1$ across the decision boundary to generate an adversarial node $\hat{\mathbf{v}}_i^1$, then we have other nodes $\mathbf{v}_j^1$ to make $M(\mathbf{v}_j^1)$ and $\mathbf{v}_k^2$ become more similar than $M(\hat{\mathbf{v}}_i^1)$ and $\mathbf{v}_k^2$,

Algorithm 1 Bandwidth Matrix Estimation

```
Input: Graph $G^1 = (V^1, E^1)$, parameter $0 < s < 1$, initial bandwidth $b_0$, and parameter $c$.
Output: Bandwidth matrix $\mathbf{B}$.
 1: Initialize all $b_1, ..., b_{N^1}$ with $b_0$
 2: for each $j \in \{1, ..., N^1\}$ do
 3:     do
 4:         Estimate the derivative $\frac{\partial f(v^1)}{\partial b_j}$ and variance $Var(\frac{\partial f(v^1)}{\partial b_j})$in Eqs.(1.6)-(1.7)
 5:         Compute the threshold $\tau_j = \sqrt{2 \cdot Var(\frac{\partial f(v^1)}{\partial b_j}) \cdot \log(cN^1)}$
 6:         if $|\frac{\partial f(v^1)}{\partial b_j}| > \tau_j$ then
 7:             $b_j = b_j s$
 8:         end if
 9:     while $|\frac{\partial f(v^1)}{\partial b_j}| > \tau_j$
10: end for
11: Return B
```

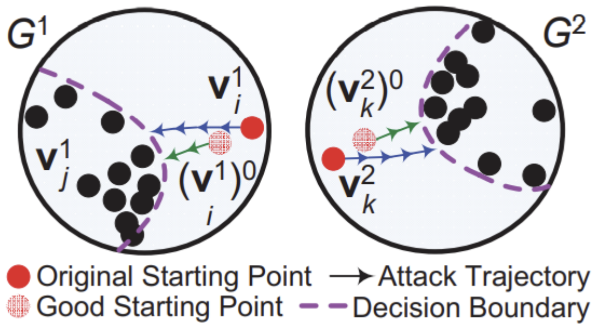


Figure 2.2: Effective Attacks

and thus a wrong matching $(\mathbf{v}_j^1, \mathbf{v}_k^2)$ will be produced. Blue and green polylines denote attack trajectories starting from original and good starting pints with gradient descent method respectively. A shortcut from good starting points $(\mathbf{v}_i^1)^0$ or $(\mathbf{v}_k^2)^0$ is able to cross the peak of the decision boundary and converge quickly, while the trajectories from the original nodes $\mathbf{v}_i^1$ or $\mathbf{v}_k^2$ take long walks to cross the non-peak boundary.

Based on the attack loss in Eq.(2.9), we propose to integrate meta learning and PGD into an MLPGD model, to produce more effective adversarial nodes with good starting points towards graph matching.

$$\begin{aligned}(\mathbf{v}_i^1)^{(t+1)} &= \Pi_{\triangle_i^1}\mathrm{sgn}\big[\mathrm{ReLU}\big(\nabla_{(\mathbf{v}_i^1)^t}\mathcal{L}((\mathbf{v}_i^1)^t,(\mathbf{v}_k^2)^t)\big)\big]\\ (\mathbf{v}_k^2)^{(t+1)} &= \Pi_{\triangle_k^2}\mathrm{sgn}\big[\mathrm{ReLU}\big(\nabla_{(\mathbf{v}_k^2)^t}\mathcal{L}((\mathbf{v}_i^1)^t,(\mathbf{v}_k^2)^t)\big)\big],\quad t=1,\cdots,T\end{aligned} \tag{2.10}$$

where $(\mathbf{v}_i^1)^t$ and $(\mathbf{v}_k^2)^t$ denotes the adversarial nodes of $\mathbf{v}_i^1$ and $\mathbf{v}_k^2$ derived at step t. ϵ specifies the budget of allowed perturbed edges for each attacked node. $\triangle_i^1 = \{(\delta_i^1)^t|\mathbf{1}^T(\delta_i^1)^t \leq \epsilon, (\delta_i^1)^t \in \{0,1\}^{N^1}\}$, where $(\delta_i^1)^t = \|(\mathbf{v}_i^1)^t - \mathbf{v}_i^1\|_2^2$, represents the constraint set of the projection operator Π, i.e., it encodes whether an edge of $\mathbf{v}_i^1$ is modified or not. $\triangle_k^2$ has the similar definition for $\mathbf{v}_k^2$. The composition of the ReLU and sign operators guarantees $(\mathbf{v}_i^1)^t \in \{0,1\}^{N^1}$ and $(\mathbf{v}_k^2)^t \in \{0,1\}^{N^2}$, as it adds (or removes) an edge or keeps it unchanged when an derivate in the gradient is positive (or negative). The outputs $(\mathbf{v}_i^1)^T$ and $(\mathbf{v}_k^2)^T$ at final step T are used as the adversarial nodes $\hat{\mathbf{v}}_i^1$ and $\hat{\mathbf{v}}_k^2$.

Searching for attack starting points for each $(\mathbf{v}_i^1, \mathbf{v}_k^2)$ in large graphs is computationally inefficient. Meta learning techniques aim to train a general model with general parameters that can quickly adapt to a variety of new learning tasks with refined parameters [145, 134, 161, 174]. This offers a great opportunity to find good attack starting points $(\mathbf{v}_i^1)^0$ and $(\mathbf{v}_k^2)^0$ for all $(\mathbf{v}_i^1, \mathbf{v}_k^2) \in D$ with lower cost, such that the generated $\hat{\mathbf{v}}_i^1$ and $\hat{\mathbf{v}}_k^2$ by the PGD model can maximize the attack loss $\mathcal{L}_D$ in Eq.(2.9).

Algorithm 2 Meta Learning-based Projected Gradient Descent (MLPGD)

Input: Batches $D_1, ..., D_C$ in a set D of node pairs, initial general policy parameters $\{\theta^1, \theta^2\}$, adaptation step size α, meta step size β
Output: Optimized $\{\theta^1, \theta^2\}$.

```
1: do
2:     Sample $C$ batches of anchor node pairs $D_1, ..., D_C$
3:     for each $c \in \{1, ..., C\}$ do
4:         Estimate gradient $\mathbf{e}_c = e(D_c, \{\theta^1, \theta^2\})$
5:         Compute adapted parameters $\{\theta_c^1, \theta_c^2\} = \{theta^1, \theta^2\} + \alpha\mathbf{e}_c$
6:     end for
7:     Update parameters $\{\theta^1, \theta^2\} = \{\theta^1, \theta^2\} + \frac{\beta}{C}\sum_{c=1}^{C} e(D_c, \{\theta_c^1, \theta_c^2\})$
8: while Not Converged
9: Return $\{\theta^1, \theta^2\}$
```

Algorithm 3 Gradient Estimation

Input: Batch D_c, general parameters $\{\theta^1, \theta^2\}$, number of samples N in Monte Carlo REINFORCE, smoothing parameter λ

Output: Gradient estimation of a.

```
1: Sample $N$ i.i.d Gaussian matrices $g_1, ..., g_N \sim N(0, \mathbf{I})$
2: **Return** gradient estimation $\frac{1}{N\lambda} a(D_c, \{\theta^1, \theta^2\} + \lambda g_i) g_i$
```

Algorithm 4 Adversarial Attack

Input: Batch D_c, perturbation budget ϵ, specific parameters $\{\theta^1, \theta^2\}$

Output: Attack loss $\mathcal{L}_{D_C}$ on D_c

```
1: $\mathcal{L}_{D_C} = 0$
2: **for each** $(\mathbf{v}_i^1, \mathbf{v}_k^2) \in D_c$ **do**
3:     Generate attack starting points $(\mathbf{v}_i^1)^0 = h^1(\mathbf{v}_i^1|\theta_c^1)$ and $(\mathbf{v}_k^2)^0 = h^2(\mathbf{v}_k^2|\theta_c^2)$
4:     Utilize PGD attack to generate adversarial nodes $(\mathbf{v}_i^1)^T$ and $(\mathbf{v}_k^2)^T$ in Eq.(1.10)
5:     Aggregate attack loss $\mathcal{L}_{D_c} += \mathcal{L}((\mathbf{v}_i^1)^T, (\mathbf{v}_k^2)^T)$ in Eq.(1.9)
6: **end for**
7: **Return** $\mathcal{L}_{D_c}$
```

Algorithm 2 presents the pseudo code of our MLPGD model. D is partitioned into C batches $D_1, \cdots, D_C$, each with equal size of $|D|/C$. The search process on each batch D_c $(1 \leq c \leq C)$ is treated as a single task, which aims to find good $(\mathbf{v}_i^1)^0$ and $(\mathbf{v}_k^2)^0$ for D_c to maximize the attack loss $\mathcal{L}_{D_c} = \sum_{(\hat{\mathbf{v}}_i^1, \hat{\mathbf{v}}_k^2) \in D_c} \mathcal{L}(\hat{\mathbf{v}}_i^1, \hat{\mathbf{v}}_k^2)$. A general model that has general parameters θ^1, θ^2 is learnt to quickly adapt to search tasks on multiple batches. The learnt θ^1, θ^2 should be sensitive to changes of each D_c, such that small changes in θ^1, θ^2 will produce high rise on $\mathcal{L}_{D_c}$ over any of $D_1, \cdots, D_C$. Line 4 estimates the gradient of $\mathcal{L}_{D_c}$ by calling Algorithm 3. In Line 5, when adapting to the task on a new D_c, θ^1, θ^2 become specific parameters θ_c^1, θ_c^2 for D_c. Here, we use $\{\theta_c^1, \theta_c^2\}$ to denote the concatenation matrix of θ_c^1 and θ_c^2. The parameters are trained by maximizing the attack loss $a(D_c, \{\theta_c^1, \theta_c^2\})$ w.r.t. general parameters θ^1, θ^2 across batches. The meta objective is given below.

$$\max \mathcal{L}_{D_c} = \max_{\theta^1, \theta^2} \sum_{c=1}^{C} a(D_c, \{\theta_c^1, \theta_c^2\}) = \sum_{c=1}^{C} a(D_c, \{\theta^1, \theta^2\} + \alpha \mathbf{e}_c) \tag{2.11}$$

In Line 7, the meta optimization is performed over the general θ^1, θ^2, while the objective is computed using the specific θ_c^1, θ_c^2. The general θ^1, θ^2 are updated in terms of the attack loss on each batch.

$$\{\theta^1, \theta^2\} = \{\theta^1, \theta^2\} + \frac{\beta}{C} \sum_{c=1}^{C} e\big(D_c, \{\theta_c^1, \theta_c^2\}\big) \tag{2.12}$$

Algorithm 4 exhibits the adversarial attack module $a\big(D_c, \{\theta_c^1, \theta_c^2\}\big)$ on a batch D_c ($1 \leq c \leq C$). In Line 3, two neural networks h^1 and h^2 with specific parameters θ_c^1 and θ_c^2 are designed to generate the attack starting points $(\mathbf{v}_i^1)^0$ and $(\mathbf{v}_k^2)^0$ of each $(\mathbf{v}_i^1, \mathbf{v}_k^2) \in D_c$. The last layers of h^1 and h^2 use the composition of the ReLU [171] and Softsign [147] as activation function to ensure $(\mathbf{v}_i^1)^0 \in \{0,1\}^{N^1}$ and $(\mathbf{v}_k^2)^0 \in \{0,1\}^{N^2}$. In Line 4, the PGD attack in Eq.(2.10) is utilized to generate the adversarial nodes $\hat{\mathbf{v}}_i^1$ and $\hat{\mathbf{v}}_k^2$. Line 5 calculates the attack loss $\mathcal{L}_{D_c}$ on D_c to provide task-specific feedback.

Standard meta learning models utilizes gradient ascent/descent techniques to compute the updated parameters on new tasks [145, 134, 161, 174]. However, the attack module in Algorithm 4 is non-smooth and non-differential w.r.t. parameters θ^1, θ^2, θ_c^1, and θ_c^2, since the perturbation is a multi-step process as well as the projection at each step is non-differential. Therefore, Algorithm 3 is proposed to employ Gaussian smoothing technique to approximate a smoothed attack module.

$$\begin{aligned} \hat{a}\big(D_c, \{\theta^1, \theta^2\}\big) &\approx (2\pi)^{-\frac{d}{2}} \int a\big(D_c, \{\theta^1, \theta^2\} + \lambda \mathbf{g}\big) \exp\big(-\frac{1}{2}\|\mathbf{g}\|_2^2\big) d\mathbf{g} \\ &= \mathbb{E}_{\mathbf{g} \sim \mathcal{N}(0,\mathbf{I})} a\big(D_c, \{\theta^1, \theta^2\} + \lambda \mathbf{g}\big) \end{aligned} \tag{2.13}$$

where $\hat{a}$ is the Gaussian smoothing of a and differentiable everywhere. λ is a smoothing parameter, and d is the number of entries in $\{\theta^1, \theta^2\}$. $\mathbf{g} \sim \mathcal{N}(0, \mathbf{I})$ that has the same size as $\{\theta_c^1, \theta_c^2\}$ is interpreted as policy exploration directions, i.e., as perturbations in policy space to be explored. Thus, the policy perturbations in $\mathbf{g}$ are introduced to θ_c^1 and θ_c^2

Table 2.1: Experiment Datasets

Dataset	**AS**		**SNS**		**DBLP**	
Graph	**v1**	**v2**	**Last.fm**	**LiveJournal**	**2013**	**2014**
#Nodes	10,900	11,113	5,682	17,828	28,478	26,455
#Edges	31,180	31,434	23,393	244,496	128,073	114,588
#Matched Nodes	6,462		2,138		4,000	

respectively. $\hat{a}$ is obtained by perturbing a at a given point along Gaussian directions and averaging the evaluations. And then, Algorithm 3 estimates the gradient of $\hat{a}$ via Monte Carlo REINFORCE method [195].

$$\begin{aligned}
&e\big(D_c, \{\theta^1, \theta^2\}\big) \approx \nabla_{\theta^1,\theta^2}\hat{a}\big(D_c, \{\theta^1, \theta^2\}\big) \approx (2\pi)^{-\frac{d}{2}} \int a\big(D_c, \{\theta^1, \theta^2\} + \lambda \mathbf{g}\big) \exp\big(-\frac{1}{2}\|\mathbf{g}\|_2^2\big)\mathbf{g} d\mathbf{g} \\
&= \frac{1}{\lambda}\mathbb{E}_{\mathbf{g}\sim\mathcal{N}(0,\mathbf{I})} a\Big(D_c, \{\theta^1, \theta^2\} + \lambda \mathbf{g}\Big)\mathbf{g} \approx \frac{1}{N\lambda}\sum_{i=1}^{N} a\big(D_c, \{\theta^1, \theta^2\} + \lambda \mathbf{g}_i\big)\mathbf{g}_i,\ \mathbf{g}_i \sim \mathcal{N}(0, \mathbf{I})
\end{aligned} \tag{2.14}$$

2.5 Experimental Evaluation

In this section, we will show the effectiveness of the GMA model in this work for deep graph matching tasks over three groups of datasets: social networks (SNS) [207], autonomous systems (AS) [133], and DBLP coauthor graphs [3], as shown in Table 4.1.

Baselines. We compare the GMA model with six state-of-the-art graph attack models. Random Attack randomly adds and removes edges to generate perturbed graphs. RL-S2V [18, 132] generates adversarial attacks on graph data based on reinforcement learning. Meta-Self [215] is a poisoning attack model for node classification by using meta-gradients to solve the bilevel optimization problem. CW-PGD [88] developed a PGD topology attack to attack a predefined or a retrainable GNN. GF-Attack [135] attacks general learning methods by devising new loss and approximating the spectrum. The majority of existing efforts focus on adversarial attacks on single graph learning. To our best knowledge, there are no other

Table 2.2: Mismatching rate (%) with 5% perturbed edges

	AS			**SNS**			**DBLP**		
Attack Model	**SNNA**	**CrossMNA**	**DGMC**	**SNNA**	**CrossMNA**	**DGMC**	**SNNA**	**CrossMNA**	**DGMC**
Clean	53.9	46.6	34.7	45.2	50.4	41.6	56.1	51.9	63.2
Random	57.5	49.9	37.6	48.8	52.0	46.8	59.8	54.0	68.8
RL-S2V	56.5	51.8	36.5	51.3	53.2	45.8	62.6	56.7	69.3
Meta-Self	63.1	55.1	45.0	55.1	64.8	51.3	65.7	63.7	73.3
CW-PGD	61.7	59.1	49.6	54.9	63.0	49.6	68.7	66.6	75.4
GF-Attack	57.9	53.7	39.5	52.9	59.6	47.9	64.9	61.1	69.1
CD-ATTACK	59.0	51.7	42.7	54.0	59.8	50.2	64.0	61.8	72.0
GMA	**64.2**	**62.9**	**54.9**	**61.2**	**69.6**	**55.7**	**74.2**	**74.3**	**80.7**

attack baselines on graph matching available. We replace the original losses in the baselines with the matching loss for fair comparison in the experiments.

Variants of GMA model. We evaluate four variants of GMA to show the strengths of different components. GMA-KDE only uses the KDE and density maximization to generate imperceptible attacks. GMA-PGD only utilizes the basic PGD [166] to produce effective attacks. GMA-MLPGD employs our proposed MLPGD model to well choose good attack starting points in the PGD. GMA operates with the full support of both KDE and MLPGD components.

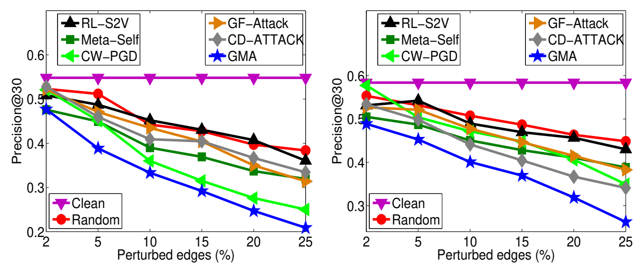
Graph matching algorithms. We validate the effectiveness of the above attack models with three representative deep graph matching methods. SNNA [49] is an adversarial learning framework to solve the weakly-supervised identity matching problem by minimizing the distribution distance. CrossMNA [15] is a cross-network embedding-based supervised network alignment method by learning inter/intra-embedding vectors for each node and by computing pairwise node similarity scores across networks. Deep graph matching consensus (DGMC) [27] is a supervised graph matching method that reaches a data-driven neighborhood consensus between matched node pairs.

Evaluation metrics. We use two popular measures in graph matching to verify the attack quality: *Accuracy* [99, 140, 205] and *Precision*@K [105, 15, 206]. A larger mismatching rate (i.e., 1 - *Accuracy* on test data) or a smaller *Precision*@K shows a better attack. K is fixed to 30 in all tests.

Attack performance on various datasets with different matching algorithms. Table 3.2 exhibits the mismatching rates of three deep graph matching algorithms on test data by eight attack models over three groups of datasets. We randomly sample 10% of known matched node pairs as training data and the rest as test data. For all attack models, the number of perturbed edges is fixed to 5% in these experiments. It is observed that among eight attack methods, no matter how strong the attacks are, the GMA method achieve the highest mismatching rates on perturbed graphs in most experiments, showing the effectiveness of GMA to the adversarial attacks. Compared to the graph matching results under other attack models, GMA, on average, achieves 21.3%, 18.8%, and 19.2% improvement of mismatching rates on AS, SNS, and DBLP respectively. In addition, the promising performance of GMA with all three graph matching models implies that GMA has great potential as a general attack solution to other graph matching methods, which is desirable in practice.

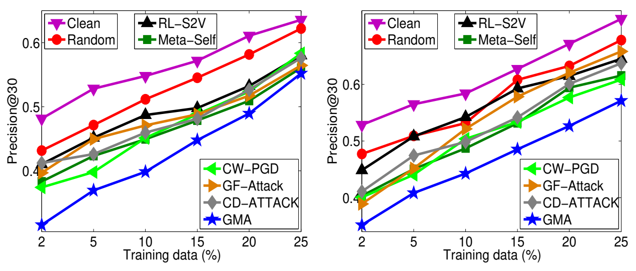
Attack performance with varying perturbation edges. Figure 2.3 presents the graph matching quality under eight attack models by varying the ratios of perturbed edges from 2% to 25%. It is obvious that the attacking performance improves for each attacker with an increase in the number of perturbed edges. This phenomenon indicates that current deep graph matching methods are very sensitive to adversarial attacks. GMA achieves the lowest $Precision$ values (< 0.488), which are still better than the other seven methods in most tests. Especially, when the perturbation ratio is large than 10%, the $Precision$ values drop quickly.

Impact of training data ratios. Figure 2.4 shows the quality of two graph matching algorithms on SNS by varying the ratio of training data from 2% to 25%. Here, the number of perturbed edges is fixed to 5%. We make the following observations on the performances by eight attack models. (1) The performance curves keep increasing when the training data ratio increases. (2) GMA outperforms other methods in most experiments with the lowest $Precision$: < 0.482 with SNNA and < 0.571 with DGMC respectively. Even when there



(a) SNNA on SNS (b) DGMC on SNS

Figure 2.3: Precision with varying perturbed edges



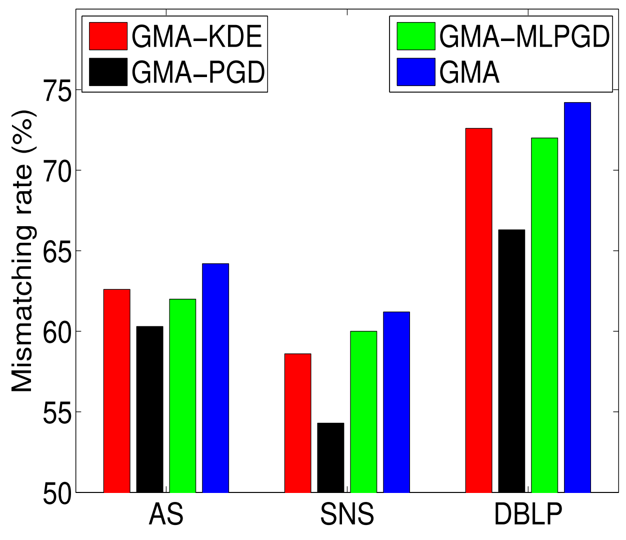
(a) SNNA on SNS (b) DGMC on SNS

Figure 2.4: Precision with varying training ratios

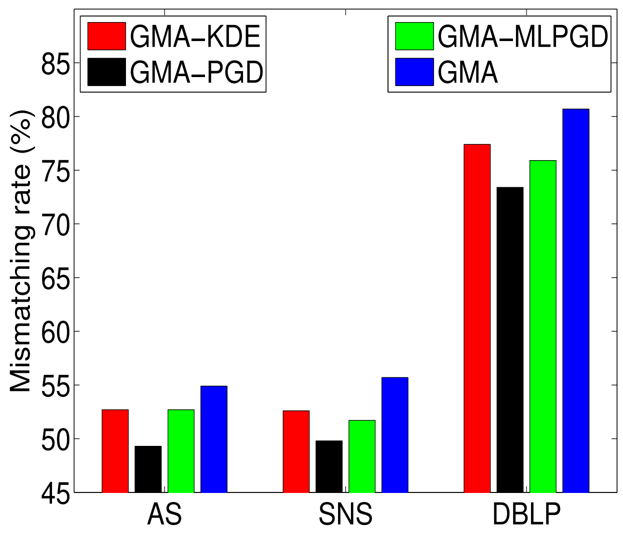
are many training data available ($\geq 20\%$), the quality degradation by GMA is still obvious, although more training data makes the graph matching models be resilient to poisoning attacks under a small perturbation budget.

Ablation study. Figure 3.5 presents the mismatching rates of graph matching on SNS with four variants of the GMA attack model. We have observed the complete GMA achieves the highest mismatching rates (¿ 54.9%) on AS, (¿ 55.7%) over SNS, and (¿ 74.2%) on DBLP, which are obviously better than other versions. Notice that GMA-MLPGD performs quite well in most experiments, compared with GMA-PGD. A reasonable explanation is that searching from good attack starting points can help the MLPGD converge quickly by crossing the peak of the decision boundary. In addition, GMA-KDE achieves the better attack performance than GMA-MLPGD. A rational guess is that it is difficult to correctly match two nodes results when they lie in dense regions with many similar nodes, although the main goal of KDE is to generate imperceptible attacks. These results illustrate both KDE and MLPGD models are important in producing effective attacks in graph matching.

Impact of perturbation budget ϵ. Figure 4.3 (a) measures the performance effect of ϵ in the MLPGD model for the graph matching by varying ϵ from 1 to 5. It is observed that when increasing ϵ, the *Precision* of the GMA model decreases substantially. This demonstrates that it is difficult to train a robust graph matching model under large ϵ constraint. However, a large ϵ can be easily detected by humans or by defender programs. Notice that

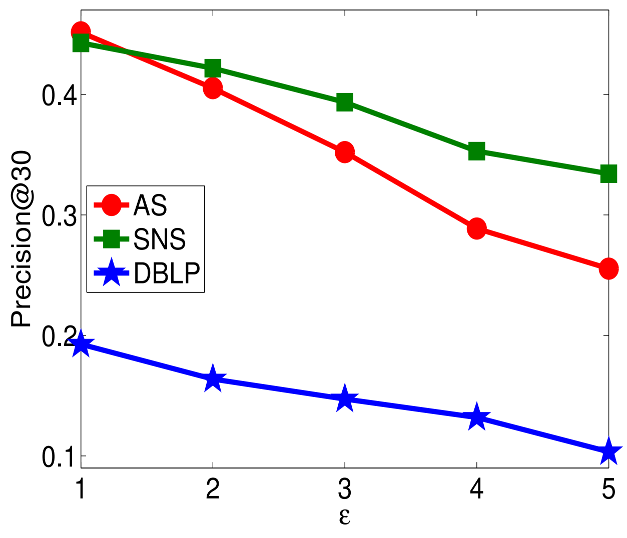


(a) SNNA

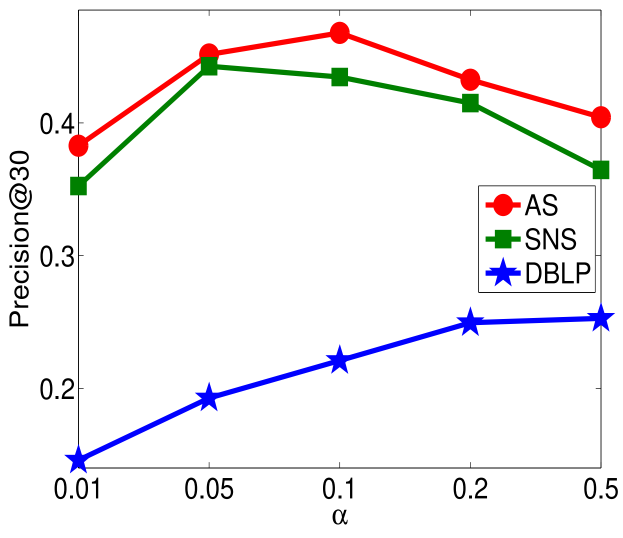


(b) DGMC

Figure 2.5: Mismatching rate (%) of GMA variants



(a) Perturbation budget ϵ



(b) Meta step size α

Figure 2.6: Precision with varying parameters

the average node degree of three groups of datasets is between 2.9 and 13.9. Thus we suggest generating both imperceptible and effective attacks for the graph matching task under ϵ between 2 and 3, such that ϵ is smaller than the average node degree.

Time complexity analysis The complexity of meta learning is $O(d^2)$, where d is the problem dimension. In the context of graph matching, it is the number of nodes in two graphs (N^s, $s = 1$ or 2). Both density estimation and PGD have complexity of $O((N^s)^2)$. Thus, the overall complexity is $O((N^s)^2)$, which is the same as most existing attack methods that search the entire graphs to find weak edges to attack.

Impact of meta step size α. Figure 4.3 (b) shows the impact of α in our MLPGD model over three groups of datasets. The performance curves initially raise when α increases. Intuitively, the MLPGD with large α can help the meta learning converge quickly. Later on, the performance curves keep relatively stable or even decreasing when α continuously increases. A reasonable explanation is that the too large α makes the meta learner take a big walk with rapid pace, such that it may miss the optimal meta parameters. Thus, it is important to determine the optimal α for the MLPGD model.

2.6 Related Work

Adversarial Attacks on Graph Data. Several recent studies have presented that graph learning models, especially deep learning-based models, are highly sensitive to adversarial attacks, i.e., carefully designed small deliberate perturbations in graph structure and attributes can cause the models to produce incorrect prediction results [185, 201, 190, 164, 198, 196, 217]. The current graph adversarial attack techniques mainly fall into two categories in terms of the attack surface: (1) evasion attacks occur after the target model is well trained in clean graphs, i.e., the learned model parameters are fixed during evasion attacks. The attacker tries to evade the graph learning models by generating malicious samples during testing phase [18, 216]; and (2) poisoning attacks, known as contamination of the training data, take place during the training time of deep learning models. An adversary tries to poison the training data by injecting carefully designed examples to cause failures of the target model on some given test samples [216, 185, 215, 9]. Since transductive learning is widely used in most graph analysis tasks, the test samples (but not their labels) are participated in the training stage, which leads to the popularity of poisoning attacks. Various adversarial attack models have been developed to show the vulnerability of graph learning models in node classification [216, 190, 88, 215, 187, 23, 186], community detection [139, 194, 137, 160], network embedding [136, 9, 135], link prediction [212], similarity search [141], malware detection [151], and knowledge graph embedding [201].

Graph Matching. Graph data analysis has attracted active research in the last decade [114, 115, 13, 109, 14, 110, 111, 74, 119, 112, 118, 120, 48, 122, 124, 123, 127, 125, 84, 85, 128]. Graph matching is one of the most important research topics in the graph domain, which aims to match the same entities (i.e., nodes) across two or more graphs and has been a heated topic in recent years [207, 179, 165, 167, 105, 15]. Research activities can be classified into three broad categories. (1) Topological structure-based techniques, which rely on only the structural information of nodes to match multiple or two input networks, including IONE [162], GeoAlign [165], Low-rank EigenAlign [172], FRUI-P [213], CrossMNA [15], MOANA [100], GWL [87], MSUIL [158], and DeepMGGE [146]; (2) Structure and/or attribute-based approaches, which utilize highly discriminative structure and attribute features for ensuring the matching effectiveness, such as FINAL [99, 204], ULink [169], CAlign [140], MASTER [181], gsaNA [91], CoLink [209], REGAL [35], UUIL [157], SNNA [49], RANA [69], CENALP [21], ORIGIN [206], and OPTANE [168]; (3) Heterogeneous methods employ heterogeneous structural, content, spatial, and temporal features to further improve the matching performance, including Factoid Embedding [197], HEP [208], LHNE [193], and DPLink [144]. Several papers review key achievements of graph matching across online information networks including state-of-the-art algorithms, evaluation metrics, representative datasets, and empirical analysis [179, 149, 152, 189].

2.7 Conclusions

In this work, we have studied the graph matching adversarial attack problem. First, we proposed to utilize kernel density estimation technique to estimate and maximize the densities of attacked nodes and generate imperceptible perturbations, by pushing attacked nodes to dense regions in two graphs. Second, we developed a meta learning based projected gradient descent method to well choose attack starting points and improve the search performance of PGD for producing effective perturbations. The GMA model achieves superior attack performance against several representative attack models.

Chapter 3

Expressive 1-Lipschitz Neural Networks for Robust Multiple Graph Learning against Adversarial Attacks

3.1 Introduction

Multiple graph learning aims to automatically extract, manage, infer, and transfer knowledge in multiple graph data. Popular multiple graph learning tasks include graph classification [70, 83, 103, 58, 65, 55, 56], graph matching (i.e., network alignment) [99, 35, 49, 15, 100, 87, 21, 38, 27, 94, 95, 68, 89], multi-graph clustering [54, 81, 24, 53, 80], multi-view network embedding [54, 67, 51, 29, 75, 28], and graph kernel [90, 5, 43, 78, 62].

We have witnessed various adversarial defense techniques to improve the robustness of single graph learning tasks against adversarial attacks, such as node classification [132, 60, 88, 77, 23, 104, 106, 42, 26, 22, 101], network embedding [19], graph clustering [39], link prediction [107], and influence maximization [52]. However, there is still a paucity of robust multiple graph learning methods under adversarial attacks, which is much more difficult to study, since the multiple graph learning tasks need to analyze both intra-graph and inter-graph links of multiple graphs. In addition, the defense strategies for single graph learning models may not work well for multiple graphs with unique characteristics, such as size, density, and degree distribution. Only recently, researchers have started to study how to improve the robustness of deep multiple graph learning methods, including graph classification [98, 93, 40, 31], graph matching [96], and multiple network embedding [129]. However, the above techniques often defend specific attacks on particular learning tasks (e.g., only graph classification or graph matching). Can we design an attack-agnostic graph-adaptive neural architecture for protecting deep multiple graph learning models from adversarial attacks?

Recently, Lipschitz-constrained neural networks are proposed to offer attack-agnostic defense solutions by imposing a Lipschitz constraint on each layer to restrict the diffusion of input perturbations on the neural networks [16, 79, 25]. The Lipschitz bound for the entire neural network is the product of the bound on each layer. This allows to constraint the change of its output in proportion to the change in its input. Lipschitz-constrained neural networks are very useful for defending multiple graph learning models since small input perturbations can be propagated within and across graphs, which dramatically amplifies the perturbations in the output space. However, bounding the Lipschitz constant and maintaining the expressive power are often regarded as orthogonal techniques with different optimization goals. Three recent studies of GroupSort [6, 17] and BCOP [50] improve the expressive power of 1-Lipschitz neural networks while enhancing the robustness by enforcing both weight norm and gradient norm as 1. We argue that simply limiting the above two norms to 1 still sacrifices the expressive power, compared with regular neural networks that do not hold the constraints on the weight and gradient. In addition, a 1-Lipschitz neural network with fixed weight and gradient norms may lead to sub-optimal defense when tackling multiple graphs with individual characteristics.

To our best knowledge, this work is the first attack-agnostic graph-adaptive 1-Lipschitz neural network for improving the robustness of deep multiple graph learning while achieving remarkable expressive power, by making the weight and gradient norms adaptive to multiple input graphs and restricting the diffusion of any input perturbations on the neural networks.

Popular 1-Lipschitz activation functions, e.g. ReLU, Sigmoid, and tanh, must trade nonlinear processing for gradient norm preservation, leading to less expressive networks. In statistics, the Weibull distribution can model hazard functions that are monotonically decreasing, increasing, or constant of the proportion of adopters over time, allowing it to describe any phase of an item's lifetime [82]. The major advantage of Weibull analysis is that it is suitable to reliability and failure analysis. In the context of robust deep multiple graph learning, the perturbation diffusion over the layers is similar to a monotonically decreasing hazard function, i.e, the attack failure possibility decreases with the perturbation diffusion. Motivated by this, a K_l-Lipschitz Weibull activation function $\bar{f}$, i.e., $\|\nabla \bar{f}(\mathbf{x})\| = K_l$, is designed to restrict the gradient norm as K_l at layer l of the neural network. In addition, we utilize nearest matrix orthogonalization and polar decomposition techniques [8, 30, 37] to discover a weight matrix $\bar{\mathbf{W}}_l$ with the norm $1/K_l$ near to the original weight $\mathbf{W}_l$, i.e., $\|\bar{\mathbf{W}}_l\| = 1/K_l$.

By enforcing $\|\nabla \bar{f}(\mathbf{x})\| = K_l$ and $\|\tilde{\mathbf{W}}_l\| = 1/K_l$ at each layer, the composite Lipschitz constant of the entire neural network is constrained to 1. We theoretically derive an important property of our 1-Lipschitz neural network for expressive and robust multiple graph learning: K_l is relevant to and should be adaptive to input graphs and layers. Given an error budget between our 1-Lipschitz neural network and regular neural network without constrained weight and gradient, we validate the existence of feasible K_l under the 1-Lipschitz constraint, i.e., derive lower and upper bounds of feasible K_l for expressive and robust multiple graph learning against adversarial attacks. The theoretical analysis is conducted to demonstrate that our 1-Lipschitz neural network with K_l-Lipschitz Weibull activation function $\bar{f}$ is universal Lipschitz function approximator, i.e., $\bar{f}$ can approximate any linear or nonlinear functions.

Empirical evaluation over graph classification and graph matching demonstrates the superior performance of our ERNN model against state-of-the-art robust graph learning models and Lipschitz-bound neural architectures. We validate that the proposed robust learning strategies are transferable to other popular graph learning tasks in Appendix.

3.2 Background and Problem Statement

3.2.1 C-Lipschitz Functions

A function $F : \mathbb{R}^N \mapsto \mathbb{R}^M$ is globally Lipschitz continuous on variable space $\mathcal{X} \subseteq \mathbb{R}^N$ if there exists a nonnegative constant $C \geq 0$ such that for all $\mathbf{x}_1$ and $\mathbf{x}_2$ in $\mathcal{X}$.

$$\|F(\mathbf{x}_2) - F(\mathbf{x}_1)\| \leq C\|\mathbf{x}_2 - \mathbf{x}_1\|, \ \forall \mathbf{x}_1, \mathbf{x}_2 \in \mathcal{X} \tag{3.1}$$

where the smallest such C for which the above inequality holds is the Lipschitz constant of F. If the Lipschitz constant of a function is C, it is called a C-Lipschitz function. If F is everywhere differentiable then its Lipschitz constant is bounded by the operator norm of its Jacobian.

If $\mathbf{x}_2$ is denoted as a perturbation of $\mathbf{x}_1$, i.e., $\mathbf{x}_2 = \mathbf{x}_1 + \delta$, then the Lipschitz constant is the maximum ratio between perturbations $\|F(\mathbf{x}_1 + \delta) - F(\mathbf{x}_1)\|$ in the output space and perturbations $\|(\mathbf{x}_1 + \delta) - \mathbf{x}_1\|$ in the input space. Thus, it is a useful metric to measure the sensitivity of the function F regarding input perturbations.

3.2.2 Lipschitz-Constrained Neural Networks

Given an input vector $\mathbf{x} \in \mathbb{R}^{N_0}$, a $(L+1)$-layer neural network $\mathbf{y} = F(\mathbf{x})$ is defined as follows: layer 0 takes $\mathbf{h}_0 = \mathbf{x}$ as input, layers $1, \cdots, L-1$ produces the hidden representations $\mathbf{h}_2, \cdots, \mathbf{h}_{L-1}$, and layer L outputs an output variable $\mathbf{y} = \mathbf{z}_L \in \mathbb{R}^{N_L}$.

$$\begin{cases} \mathbf{z}_l = \mathbf{W}_l \mathbf{h}_{l-1} + \mathbf{b}_l, \mathbf{h}_l = f(\mathbf{z}_l), \textbf{if } 1 \leq l \leq L-1, \\ \mathbf{z}_l = \mathbf{W}_l \mathbf{h}_{l-1} + \mathbf{b}_l, \mathbf{y} = \mathbf{z}_l, \textbf{if } l = L. \end{cases} \tag{3.2}$$

where N_l is the dimensionality of layer l, $\mathbf{W}_l \in \mathbb{R}^{N_l \times N_{l-1}}$ is the weight matrix between layers $l-1$ and l, and $\mathbf{b}_l \in \mathbb{R}^{N_l}$ is the bias for layer l. $\mathbf{z}_l = [\mathbf{z}_{l1}, \cdots, \mathbf{z}_{lN_l}]$ denotes the pre-activation vector in layer l and $\mathbf{h}_l = [\mathbf{h}_{l1}, \cdots, \mathbf{h}_{lN_l}]$ is the activation vector with $\mathbf{z}_l$. f is the activation function. At layer L, the pre-activation $\mathbf{z}_L$ is used as the final output $\mathbf{y}$.

The Lipschitz constant C of neural network is derived below.

$$C = \|\mathbf{W}_L\| \cdot \|\nabla_{\mathbf{z}_{L-1}} f\| \cdot \|\mathbf{W}_{L-1}\| \cdots \|\nabla_{\mathbf{z}_1} f\| \cdot \|\mathbf{W}_1\| \tag{3.3}$$

Adversarial robustness. When the function F is characterized by a deep neural network, tight bounds on its Lipschitz constant can be extremely useful to improve the robustness of the neural network against adversarial attacks. Concretely, if the Lipschitz constant C of F is limited to a small number, say 1 used in GroupSort [6, 17] and BCOP [50], such that $\|F(\mathbf{x}+\delta)-F(\mathbf{x})\| \leq \|(\mathbf{x}+\delta)-\mathbf{x}\|$ for a 1-Lipschitz neural network, then this can help effectively control the diffusion of input perturbations through the neural networks.

3.2.3 Multiple Graph Learning

Given a set of S graphs $\mathcal{G}=\{G^1,\cdots,G^S\}$. Each graph is denoted as $G^s=(V^s,E^s)$ $(1 \leq s \leq S)$, where $V^s=\{v_1^s,\cdots,v_{N^s}^s\}$ is the set of N_s nodes and $E^s=\{(v_i^s,v_j^s): 1 \leq i,j \leq N_s, i \neq j\}$ is the set of edges. Each G^s has an $N_s \times N_s$ binary adjacency matrix $\mathbf{A}^s$, where each entry $\mathbf{A}_{ij}^s=1$ if there exists an edge $(v_i^s,v_j^s) \in E^s$; otherwise $\mathbf{A}_{ij}^s=0$. $\mathbf{A}_{i:}^s$ specifies the i^{th} row vector of $\mathbf{A}^s$ and is used to denote the representation of a node v_i^s. In this paper, we focus on enhancing the robustness of two multiple graph learning tasks, but it is straightforward to extend to others.

Graph classification. We associate each graph G^s with a label $y^s \in \mathcal{Y}=\{1,2,...,Y\}$, where Y is the number of classes. The training data denotes a set of known graph-label pairs, i.e., $D=\{(G^s,y^s)|G^s \leftrightarrow y^s, G^s \in \mathcal{G}, y^s \in \mathcal{Y}\}$, where $G^s \leftrightarrow y^s$ indicates that G^s and y^s are the corresponding graph-label pair. The goal of graph classification is to employ D as the training data to predict label y^s for graph G^s in the test data. A classifier $F:\mathcal{G} \mapsto \mathcal{Y}$ is optimized to minimize the following loss over all labeled graphs.

$$\mathcal{L}=\frac{1}{|D|}\sum_{s=1}^{|D|} L\big(F(G^s),y^s\big) \tag{3.4}$$

where L is the cross-entropy loss.

Graph matching. The entire training data consists of a set of training data between pairwise graphs, i.e., $D=\{D^{12},\cdots,D^{1S},\cdots,D^{(S-1)S}\}$. Each D^{st} $(1 \leq s < t \leq S)$ specifies a set of pre-aligned node pairs $D^{st}=\{(v_i^s,v_j^t)|v_i^s \leftrightarrow v_j^t, v_i^s \in V^s, v_j^t \in V^t\}$, where $v_i^s \leftrightarrow v_j^t$

represents that two nodes v_i^s and v_j^t are the equivalent ones in two graphs G^s and G^t. The objective of graph matching is to utilize D^{st} as the training data to identify the one-to-one node matchings between nodes v_i^s and v_j^t in the test data. By following the same idea in existing efforts [59, 105, 91, 49], this paper aims to learn an embedding function F to map the node pairs $(v_i^s, v_j^t) \in D^{st}$ with different features across two graphs into common embedding space, i.e, minimize the distances between projected source nodes $F(v_i^s) \in D^{st}$ and target ones $F(v_j^t) \in D^{st}$. The node pairs $(v_i^s, v_j^t) \in D^{st}$ with the smallest distances in the test data are selected as the matching results.

$$\mathcal{L} = \sum_{s=1}^{S} \sum_{t=s+1}^{S} \mathbb{E}_{(v_i^s, v_j^t) \in D^{st}} \| F(v_i^s) - F(v_j^t) \|_2^2 \tag{3.5}$$

With the perturbed graphs as input, this paper aims to develop an attack-agnostic graph-adaptive 1-Lipschitz neural network to improve the robustness against adversarial perturbations while achieving remarkable expressive power in the context of multiple graph learning.

3.3 Expressive and Robust 1-Lipschitz Neural Network for Multiple Graph Learning

Deep learning models have demonstrated their remarkable expressive power by using nonlinear activation functions to stimulate and learn any linear or nonlinear functions representing a question, and provide accurate predictions. Regular neural networks do not hold the constraints on the weight and gradient in order to achieve the superior nonlinearity. GroupSort [6, 17] and BCOP [50] proposed 1-Lipschitz neural networks to achieve the model robustness while improving the expressive power by limiting both weight and gradient norms as 1. Their Lipschitz constant is computed below.

$$\begin{aligned} C &= \|\tilde{\mathbf{W}}_L\| \cdot \|\nabla_{\tilde{\mathbf{z}}_{L-1}} \tilde{f}\| \cdot \|\tilde{\mathbf{W}}_{L-1}\| \cdots \|\nabla_{\tilde{\mathbf{z}}_1} \tilde{f}\| \cdot \|\tilde{\mathbf{W}}_1\| \\ &= 1 \cdot 1 \cdot 1 \cdots 1 \cdot 1 = 1 \end{aligned} \tag{3.6}$$

The GroupSort activation function $\tilde{f}$ is essentially a permutation operation that sorts and permutes the elements in each $\mathbf{z}_l$ on each layer l. Thus, both $\|\tilde{\mathbf{W}}_l\|$ and $\|\nabla_{\tilde{\mathbf{z}}_l} \tilde{f}\|$ are constrained to 1. This may lead to sub-optimal defense when tackling multiple graphs with individual characteristics.

We propose an attack-agnostic graph-adaptive 1-Lipschitz neural network with a K_l-Lipschitz activation function $\bar{f}$, i.e., $\|\nabla_{\mathbf{z}_{l-1}} \bar{f}\| = K_l$ and a constrained weight matrix $\|\bar{\mathbf{W}}_l\| = 1/K_l$ for achieving better expressive power.

$$\begin{aligned} C &= \|\bar{\mathbf{W}}_L\| \cdot \|\nabla_{\bar{\mathbf{z}}_{L-1}} \bar{f}\| \cdots \|\bar{\mathbf{W}}_2\| \cdot \|\nabla_{\bar{\mathbf{z}}_1} \bar{f}\| \cdot \|\bar{\mathbf{W}}_1\| \\ &= \frac{1}{K_L} \cdot K_L \cdots \frac{1}{K_2} \cdot K_2 \cdot 1 = 1 \end{aligned} \tag{3.7}$$

where $\|\bar{\mathbf{W}}_l\| = 1/K_l$ if $l > 1$, otherwise $\|\bar{\mathbf{W}}_l\| = 1$. In this paper, we use ∞-norm for both weight and gradient. For ease representation, we use $\|\cdot\|$ to replace $\|\cdot\|_\infty$ in our 1-Lipschitz neural network.

The following theorems validate the existence of feasible K_l under the 1-Lipschitz constraint for robust and expressive multiple graph learning against adversarial attacks. Theorem 2 derives lower bound of feasible K_l and demonstrates that selecting an appropriate K_l rather than 1 can guarantee that our 1-Lipschitz neural network achieves better expressive power than GroupSort [6, 17]. Theorem 3 derives upper bound of feasible K_l when we are given an error budget between our 1-Lipschitz neural network and regular neural network without constrained weight and gradient. Theorem 3 also exhibits that K_l is relevant to and should be adaptive to input graphs and layers. Definition 1, Lemma 1, and Theorem 1 are the preparation of the proof of Theorem 2-3

Definition 3.1 (Finite Partition of an Interval) *A partition P of an interval $[a, b]$ on the real line is a sequence of a finite number of subintervals of $[a, b]$*

$$P = \{[x_0, x_1], [x_1, x_2], \cdots, [x_{m-1}, x_m], \cdots, [x_{M-1}, x_M]\} \tag{3.8}$$

where $a = x_0 < x_1 < x_2 < \cdots < x_{m-1} < x_m < \cdots < x_{M-1} < x_M = b$. The points $x_m, 0 \leq m \leq M$, are called the partition points in P. Each $[x_{m-1}, x_m]$ is referred to as a subinterval of the partition P.

Based on Definition 1, given large enough M, it is always feasible to partition an interval $[a, b]$ into multiple subintervals, such that any continuous nonlinear function f on $[a, b]$ become linear or near-linear on each subinterval.

Lemma 3.1 (Lagrange's Mean Value Theorem) *For any continuous function f on the closed interval $[a, b]$ and differentiable on the open interval (a, b), then there exists a point c in (a, b) such that the tangent at c is parallel to the secant line through the endpoints $(a, f(a))$ and $(b, f(b))$ [71].*

$$f'(c) = \frac{f(a) - f(b)}{a - b} \tag{3.9}$$

When a continuous function f is linear on the interval $[a, b]$, its slope is equal to $f'(c)$. The above observations inspire us to design and transform a nonlinear activation function $\bar{f}$ going through the origin into an approximate piecewise linear function for the proof of the following three theorems and to finally derive the lower and upper bounds of K_l.

Theorem 3.1 *For any K_l-Lipschitz nonlinear activation function $\bar{f} : \mathbb{R}^N \mapsto \mathbb{R}^N$, if $\bar{f}$ is everywhere differentiable and $\bar{f}(\mathbf{x}) = \mathbf{0} \in \mathbb{R}^N$ at $\mathbf{x} = \mathbf{0} \in \mathbb{R}^N$, then there must exist a linear function g such that $\bar{f}(\mathbf{x}) \leq g(\mathbf{x})$ for $\forall \mathbf{x}, \mathbf{x} \in \mathbb{R}^N$.*

Theorem 3.2 *By following the definition in Eq.(2), we build a $(L+1)$-layer 1-Lipschitz neural network $\bar{F} : \mathbb{R}^{N_0} \mapsto \mathbb{R}^{N_L}$, with a gradient norm preserving activation function $\|\nabla_{\bar{\mathbf{z}}_{l-1}} \bar{f}\| = K_l$ almost everywhere and a norm-constrained weight matrix $\|\bar{\mathbf{W}}_l\| = 1/K_l$ like the definitions in Eq.(7). If $K_l > 1$ for $\forall l, 2 \leq l \leq L$, our 1-Lipschitz neural network $\bar{F}$ achieves better expressive power than the neural network $\tilde{F}$ constructed by the GroupSort model [6, 17].*

Theorem 3.3 *Given a regular $(L+1)$-layer fully connected neural network $F : \mathbb{R}^{N_0} \mapsto \mathbb{R}^{N_L}$ with unconstrained ReLU as the activation and unrestricted weight, and our $(L+1)$-layer 1-Lipschitz neural network $\bar{F}$ defined in Theorem 2, if an error budget between each layer of two neural networks is limited to ε, i.e., $\|\bar{\mathbf{z}}_l - \mathbf{z}_l\| \leq \varepsilon$, where $\mathbf{z}_l$ and $\bar{\mathbf{z}}_l$ are the representations at layer l in F and $\bar{F}$ respectively, then $K_l \leq \min\Big\{\frac{\|\bar{f}(\bar{\mathbf{z}}_{l-1})\|}{\|\mathbf{W}_l f(\mathbf{z}_{l-1})\| - \varepsilon}, \max\big\{-\min_j \frac{\partial \bar{f}}{\partial \bar{\mathbf{z}}_{(l-1)j}}, \max_j \frac{\partial \bar{f}}{\partial \bar{\mathbf{z}}_{(l-1)j}}\big\}\Big\}$.*

Proof. Please refer to Appendix for detailed proof of the above three theorems.

In this paper, we use a random variable $\mathbf{x}$ denoting the node representation as input of the neural network for multiple graph learning. Since $\mathbf{h}_0 = \mathbf{x}$, $\mathbf{z}_1$ depends on $\mathbf{h}_0$, and other $\mathbf{z}_l$ ($\forall l, 2 \leq l \leq L$) are related to $\mathbf{z}_1$, the upper bound of K_l is relevant to input graphs and layers.

Based on Theorems 2-3, we need to make $1 < K_l \leq \min \Big\{\frac{\|\bar{f}(\bar{\mathbf{z}}_{l-1})\|}{\|\mathbf{W}_l f(\mathbf{z}_{l-1})\| - \varepsilon}, \max\big\{-\min_j \frac{\partial \bar{f}}{\partial \bar{\mathbf{z}}_{(l-1)j}}, \max_j \frac{\partial \bar{f}}{\partial \bar{\mathbf{z}}_{(l-1)j}}\big\}\Big\}$, such that our 1-Lipschitz neural network $\bar{F}$ achieves better expressive power than the GroupSort model $\tilde{F}$ and comparable quality to the regular network F at layer l.

3.3.1 Constraining $\|\bar{\mathbf{W}}_l\| = 1/K_l$

According to Theorem 2 in the GroupSort paper [6], when enforcing $\|\tilde{\mathbf{W}}_l\| = 1$, it needs to adjust the weight matrix $\mathbf{W}_l$ to have singular values of 1 without sacrificing nonlinear processing capacity. In our case, the equivalent problem is that all singular values of the norm-contained weight matrix $\|K_l \bar{\mathbf{W}}_l\|$ are equal to 1.

Recall that the singular values of a real matrix $\mathbf{A}$ are the eigenvalues of the positive-semidefinite real matrix $\mathbf{A}^T\mathbf{A}$, where $\mathbf{A}^T$ is the transpose of $\mathbf{A}$. The singular values of $\mathbf{A}$ are all 1 iff $\mathbf{A}$ is orthogonal, i.e., $\mathbf{A}^T\mathbf{A} = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix. Thus, the original problem is equivalent to finding the nearest orthonormal matrix of $K_l\mathbf{W}_l$.

For ease of representation, let $\mathbf{A} = K_l\mathbf{W}_l$ and $\mathbf{B} = K_l\bar{\mathbf{W}}_l$. Formally, given an $N_l \times N_{l-1}$ matrix $\mathbf{A}$, we aim to find the nearest $N_l \times N_{l-1}$ matrix $\mathbf{B}$ with N_l orthonormal columns (i.e. $\mathbf{B}^T\mathbf{B} = \mathbf{I}$), i.e., we try to minimize $\|\mathbf{A} - \mathbf{B}\|_F = \sqrt{\text{trace}\left((\mathbf{A} - \mathbf{B})^T(\mathbf{A} - \mathbf{B})\right)}$.

Polar decomposition is an effective technique that finds the nearest orthonormal matrix. However, traditional iterative algorithms have non-trival computational cost based on operation counts, including Björck Orthonormalization [8] and Newton iteration-based methods [30, 10]. In order to improve the cost of the iterative algorithms, a fast hybrid algorithm was proposed to adaptively switches from the matrix inversion based iteration to a matrix multiplication based iteration [36]. The hybrid algorithm tends to require at most 7 iterations for convergence. In addition, if $\mathbf{B}$ is not required to full accuracy then there is no need to iterate to converge–just 1 or 2 iterations may yield a sufficiently accurate approximation to $\mathbf{B}$. In our implementation, we use 3-4 iterations of the fast hybrid algorithm per forward pass to get a good approximation $\|K_l\bar{\mathbf{W}}_l\|$.

Theorem 4 exhibits the uniqueness of the nearest orthonormal matrix $\mathbf{B}$ by using the above fast hybrid algorithm.

Theorem 3.4 *Given an $N_l \times N_{l-1}$ matrix $\mathbf{A}$, the nearest orthonormal matrix $\mathbf{B}$ of $\mathbf{A}$ is unique. It is equal to $\hat{\mathbf{B}} = \mathbf{A}\mathbf{H}^{-1}$ by using the fast hybrid polar decomposition, where $\mathbf{H} = \sqrt{\mathbf{A}^T\mathbf{A}}$ is positive definite.*

Proof. Please refer to Appendix for detailed proof.

Practically, we can compute a residual $\mathbf{R} = \mathbf{A}^T\mathbf{A} - \mathbf{I}$ and then use a series to approximate $\hat{\mathbf{B}}$.

$$\begin{aligned}\hat{\mathbf{B}} &= \mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1/2} = \mathbf{A}(\mathbf{I}+\mathbf{R})^{-1/2} \\ &= \mathbf{A} - \mathbf{A}\mathbf{R}(1/2 - 3\mathbf{R}/8 + 5\mathbf{R}^2/16 - 35\mathbf{R}^4/128 + \cdots)\end{aligned} \tag{3.10}$$

In order to further improve the computational cost, we can calculate the first few terms without losing much accuracy. If $\|\mathbf{R}\|_F$ comes near to zero, then $\hat{\mathbf{B}}$ can be approximated adequately $\check{\mathbf{B}} = \mathbf{A} - \mathbf{A}\mathbf{R}/2 = \hat{\mathbf{B}}(\mathbf{I} - 3\mathbf{R}^2/8 + \mathbf{R}^3/8 - \cdots)$ as its residual $\check{\mathbf{B}}^T\check{\mathbf{B}} - \mathbf{I} = \mathbf{R}^2(\mathbf{R} - 3\mathbf{I})/4$ is trivial. On the other hand, if $\|\mathbf{R}\|_F << 1/2$, then the columns of $\check{\mathbf{B}}$ will be more nearly orthonormal than those of $\mathbf{A}$; and repeating upon $\check{\mathbf{B}}$ the process performed upon $\mathbf{A}$ can yield an approximation $\hat{\mathbf{B}}(\mathbf{I} - 27\mathbf{R}^4/128 + \cdots)$.

3.3.2 K_l-Lipschitz Weibull Activation $\|\nabla_{\bar{\mathbf{z}}_l}\bar{f}\| = K_l$

To bound the Lipschitz constant of neural network, the gradient norm must be preserved by each layer in the network during backpropagation. Unfortunately, Theorem 5 exhibits that norm-constrained neural networks with common 1-Lipschitz activation functions (e.g. ReLU, Leaky ReLU, Sigmoid, SoftPlus, or tanh) must trade nonlinear processing for gradient norm preservation, leading to less expressive networks. Namely, such norm-constrained neural networks can only approximate the linear functions with less expressive power. It is straightforward to extend the conclusion of Theorem 5 to our 1-Lipschitz neural network. In addition, for our 1-Lipschitz neural network, most of popular nonlinear activation functions with gradient norm preservation $\|\nabla_{\bar{\mathbf{z}}_{l-1}} f\| = K_l$, can not achieve the feasible K_l, since the maximum values of their derivatives are smaller than the lower bound of the feasible K_l, e.g, 1 for ReLU, Leaky ReLU, and PReLU, 0.25 for Sigmoid, 1 for tanh and Softplus, and 1 for GroupSort [6]. Therefore, we utilize the Weibull distribution to design an expressive K_l-Lipschitz nonlinear activation function to allow our 1-Lipschitz neural network to approximate any functions.

Theorem 3.5 *Consider a 1-Lipschitz neural network $\bar{F} : \mathbb{R}^{N_0} \mapsto \mathbb{R}$, built with norm-constrained weights ($\|\bar{\mathbf{W}}_l\| \leq 1$) and 1-Lipschitz, element-wise, monotonic activation functions $\|\nabla_{\bar{\mathbf{z}}_{l-1}} f\| = 1$. If $||\nabla_{\mathbf{x}} \bar{F}(\mathbf{x})|| = 1$ almost everywhere, then $\bar{F}$ is linear [6].*

In statistics, the Weibull distribution is a continuous probability distribution [82]. It can model hazard functions that are monotonically decreasing, increasing or constant of the proportion of adopters over time, allowing it to describe any phase of an item's lifetime. Therefore, the major advantage of Weibull analysis is that it is suitable to reliability and failure analysis. In addition, a recent study reports that the non-saturating nonlinear activation functions, such as ReLU and Leaky ReLU, often achieve faster training than the saturating ones, e.g., Sigmoid and tanh [46]. In order to achieve the advantage of non-saturating nonlinearity, we combine T Weibull activation functions $\bar{f}_1(z), \cdots, \bar{f}_T(z)$ with different parameters into a composite one, such that the upper bound of $\bar{f}(z)$ is increased to T.

$$\bar{f}(z) = \begin{cases} \sum_{t=1}^{T} \bar{f}_t(z), \text{if } z \geq \mu_t, \\ K_l z, \qquad \text{if } z < \mu_t. \end{cases}, f_t(z) = 1 - e^{-(\frac{z-\mu_t}{\lambda_t})^{\alpha_t}} \tag{3.11}$$

where $\bar{f}_t$ is the t^{th} Weibull activation function with unique parameters α_t, λ_t, and μ_t. z is an element in $\bar{\mathbf{z}}_l$, $\alpha_t > 0$ is the shape parameter, $\lambda_t > 0$ is the scale parameter, and μ_t is the shift parameter. A value of $\alpha_t < 1$ indicates that the failure rate decreases with time. This happens if there is significant "infant mortality", or few defective parts failing to result in the malfunction of the entire item early and the failure rate decreasing over time as more parts gradually become defective over time. In the context of robust deep multiple graph learning, the perturbation diffusion over the layers is similar to a monotonically decreasing hazard function, i.e, the attack failure possibility decreases with the perturbation diffusion, and the diffusion of any perturbations finally leads to the attack success when enough diffusion is allowed. Thus, the Weibull activation function can effectively model the relationship between the perturbation diffusion and the attack failure.

The derivative of $f(z)$ is thus generated as follows.

$$\bar{f}'(z) = \begin{cases} \sum_{t=1}^{T} \frac{\alpha_t}{\lambda_t}\Big(\frac{z-\mu_t}{\lambda_t}\Big)^{\alpha_t - 1} e^{-(\frac{z-\mu_t}{\lambda_t})^{\alpha_t}}, \text{if } z \geq \mu_t, \\ K_l, \qquad\qquad\qquad\qquad\qquad \text{if } z < \mu_t. \end{cases} \tag{3.12}$$

Theorem 3.6 *Given T Weibull activation functions with the definition in Eq.(11), there must exist solutions of parameters α_t, λ_t, and μ_t to guarantee $\|\nabla_{\bar{\mathbf{z}}_l}\bar{f}\| = K_l$.*

Proof. Please refer to Appendix for detailed proof.

In statistics, a scale parameter λ_t is a special kind of numerical parameter of a parametric family of probability distributions. If it is large, then the distribution will be more spread out; if it is small then it will be more concentrated. Therefore, $\bar{f}(z)$ is more sensitive to λ_t. Notice that $\|\nabla_{\bar{\mathbf{z}}_l}\bar{f}\|_\infty = \bar{f}'(\bar{\mathbf{z}}_{lU}) = \frac{\partial \bar{f}}{\partial \bar{\mathbf{z}}_{lU}}$, where $\frac{\partial \bar{f}}{\partial \bar{\mathbf{z}}_{lU}} = \max\{|\frac{\partial \bar{f}}{\partial \bar{\mathbf{z}}_{l1}}|, \cdots, |\frac{\partial \bar{f}}{\partial \bar{\mathbf{z}}_{lN_l}}|\}$. With selected α_t and μ_t, we utilize the Newton-Raphson method to find an approximate λ_t of the following equation to make $\bar{f}'(\bar{\mathbf{z}}_{lU}) = K_l$ when $z \geq \mu_t$ [32].

$$\frac{\alpha_t}{\lambda_t}\Big(\frac{\bar{\mathbf{z}}_{lU}-\mu_t}{\lambda_t}\Big)^{\alpha_t - 1} e^{-(\frac{\bar{\mathbf{z}}_{lU}-\mu_t}{\lambda_t})^{\alpha_t}} = K_l \tag{3.13}$$

3.3.3 Universal Approximation of K_l-Lipschitz Weibull Activation

The following theorems demonstrate that our 1-Lipschitz neural network architecture with a K_l-Lipschitz activation function $\bar{f}$ is universal Lipschitz function approximator, i.e., $\bar{f}$ can approximate any linear or nonlinear functions.

Definition 3.2 *We say that a set of functions, $\mathcal{F}$, is a lattice if for any $f, g \in \mathcal{F}$ we have* $\max(f, g) \in \mathcal{F}$ *and* $\min(f, g) \in \mathcal{F}$ *(where* max *and* min *are defined pointwise).*

Lemma 3.2 (Restricted Stone-Weierstrass Theorem) *Suppose that (X, d_X) is a compact metric space with at least two points and $\mathcal{F}$ is a lattice in $C_{\mathcal{F}}(X, \mathbb{R})$ with the property that for any two distinct elements $x, y \in X$ and any two real numbers a and b such that*

$|a-b| \leq d_X(x,y)$ there exists a function $f \in \mathcal{F}$ such that $f(x)=a$ and $f(y)=b$. Then $\mathcal{F}$ is dense in $C_{\mathcal{F}}(X,\mathbb{R})$ [6].

Theorem 3.7 *Let $\mathcal{LN}_{\infty}^{N_L} : \mathbb{R}^{N_0} \mapsto \mathbb{R}^{N_L}$ denote the class of $(L+1)$-layer 1-Lipschitz neural networks $\bar{F}$ with norm-constrained weight matrices $\|\bar{\mathbf{W}}_l\|_\infty = 1$ $(l=1)$ and $\|\bar{\mathbf{W}}_l\|_\infty = 1/K_l$ $(l>1)$, and gradient norm preserving activation function $\|\nabla_{\bar{\mathbf{z}}_{l-1}}\bar{f}\|_\infty = K_l$, by following the definitions in Eqs.(2) and (7). Let input $\mathcal{X}$ be a closed and bounded subset of $\mathbb{R}^{N_0}$ with the L_∞ metric. Then the closure of $\mathcal{LN}_{\infty}^{N_L}$ is dense in $C_{\mathcal{F}}(\mathcal{X},\mathbb{R})$.*

Proof. Please refer to Appendix for detailed proof of the above two theorems.

Theorem 7 demonstrates that the closure of $\mathcal{LN}_{\infty}^{N_L}$ is dense in $C_{\mathcal{F}}(\mathcal{X},\mathbb{R})$. Namely, for any input $\mathbf{x} \in \mathbb{R}^{N_0}$, function $\bar{F}(\mathbf{x}) \in \mathcal{LN}_{\infty}^{N_L}$ can be used to approximate any function $\mathbb{R}^{N_0} \mapsto \mathbb{R}^{N_L}$ in continuous function space $C_{\mathcal{F}}(\mathcal{X},\mathbb{R})$.

3.4 Experiments

Table 3.1: Experiment Datasets

Dataset	**AS**		**CAIDA**		**DBLP**	
Graph	G^1	G^1	G^1	G^2	**2013**	**2014**
#Nodes	10,900	11,113	16,493	16,301	28,478	26,455
#Edges	31,180	31,434	33,372	32,955	128,073	114,588
#Matched Nodes	7,943		7,884		4,000	

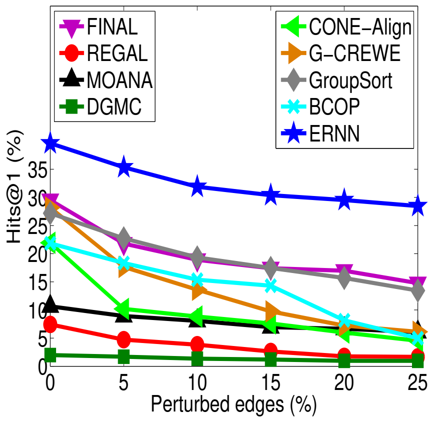
Dataset	#Graphs	#Avg. Nodes	#Avg. Edges	#Classes
BZR	405	35.75	38.36	2
BZR_MD	306	21.30	225.06	2
MUTAG	188	17.93	19.79	2

Table 3.2: Graph classification with 5% perturbed edges

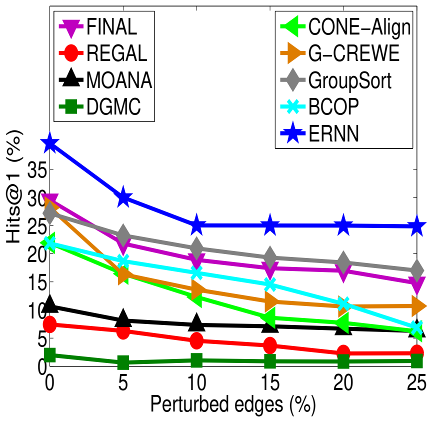
Dataset	BZR_MD		MUTAG		BZR	
Metric	**Acc.**	**RMSE**	**Acc.**	**RMSE**	**Acc.**	**RMSE**
PAN	0.541	0.680	0.681	0.570	0.779	0.471
Pro-GNN	0.631	0.610	0.643	0.601	0.738	0.512
GRAND	0.532	0.684	0.633	0.610	0.759	0.492
GCN-SVD	0.648	0.594	0.653	0.593	0.767	0.484
RoboGraph	0.667	0.579	0.644	0.601	0.790	0.458
GraphCL	0.652	0.593	0.545	0.680	0.808	0.439
GroupSort	0.547	0.676	0.606	0.636	0.756	0.494
BCOP	0.582	0.648	0.570	0.656	0.741	0.509
ERNN	**0.691**	**0.540**	**0.788**	**0.461**	**0.820**	**0.425**

We perform extensive evaluation on the robustness of our ERNN model for graph classification on three real datasets: BZR, BZR_MD and MUTAG [40, 4] and for graph matching over three datasets: autonomous systems (AS) [1], CAIDA relationships datasets [2], and DBLP coauthor graphs [3], as shown in Table 4.1.

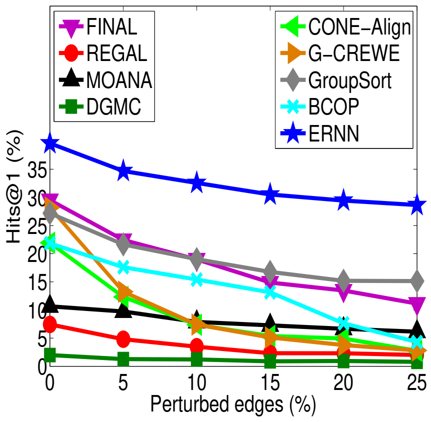
Graph classification baselines. We compare our ERNN model with one regular graph classification algorithm, two robust node classification models, one general graph denoising method, two state-of-the-art robust graph classification models against adversarial attacks, and two representative Lipschitz-bound neural architectures for restricting the perturbation propagation. **PAN** [57] is a path integral based GNN containing self-consistent convolution and pooling units for producing regular graph classification. **Pro-GNN** [42] jointly learns a clean graph and a robust GNN model for defending node classification. **GRAND** [26] is a graph random neural network with random propagation and data augmentation to increase the robustness of node classification. **GCN-SVD** [23] is a general



(a) RND Attack

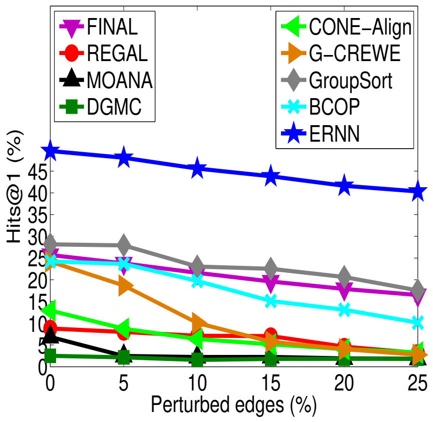


(b) NEA Attack

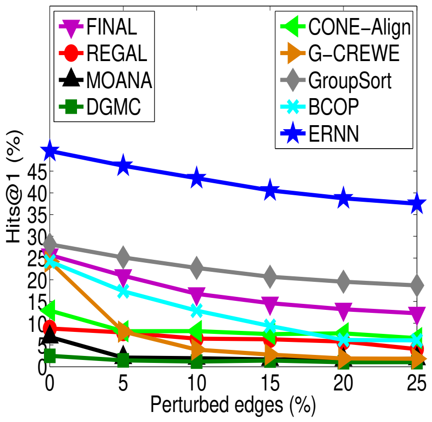


(c) GMA Attack

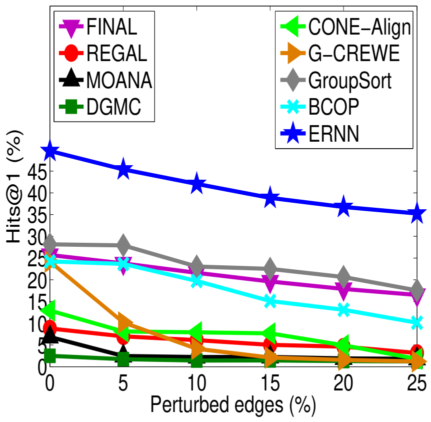
Figure 3.1: Matching on AS with varying perturbed edges



(a) RND Attack



(b) NEA Attack



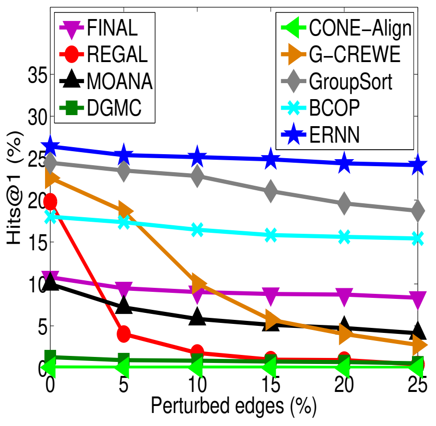
(c) GMA Attack

Figure 3.2: Matching on CAIDA with varying perturbed edges

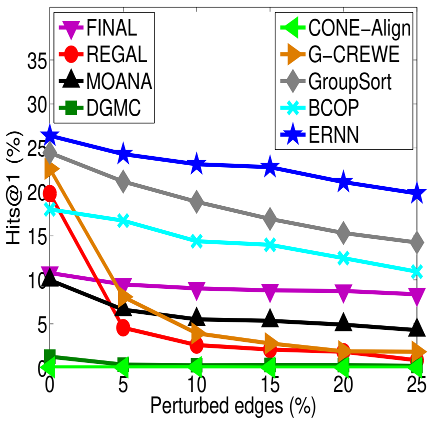
perturbation elimination model irrelevant to specific graph learning architectures. **Robo-Graph** [40] is the first certifiably robust graph classification model based on Lagrange dualization and convex envelope. **GraphCL** [93] is a graph contrastive learning framework with data augmentations for GNN pre-training for boosting the robustness of graph classification. **GroupSort** [6, 17] is a 1-Lipschitz fully-connected neural network that restricts the perturbation propagation by imposing a Lipschitz constraint on each layer. **BCOP** [50] is a Lipschitz-constrained convolutional network with expressive parameterization of orthogonal convolution operations. For two node classification models, the average of node labels within the same graphs is output as graph labels.

Graph matching baselines. We compare the ERNN model with six state-of-the-art graph matching algorithms, GroupSort, and BCOP. **FINAL** [99] leverages both node and edge attributes to solve the attributed network alignment problem. Its supervised version with prior alignment preference matrix is used for the evaluation. **REGAL** [35] is an unsupervised network alignment framework that infers soft alignments by comparing joint node embeddings across graphs. and by computing pairwise node similarity scores across networks. **MOANA** [100] is a supervised coarsening-alignment-interpolation multilevel network alignment algorithm with the supervision of a prior node similarity matrix. Deep graph matching consensus (**DGMC**) [27] is a supervised graph matching method that reaches a data-driven neighborhood consensus between matched node pairs. **CONE-Align** [11] models intra-network proximity with node embeddings and uses them to match nodes across networks in an unsupervised manner. **G-CREWE** [66] is a rapid unsupervised network alignment method via both graph compression and embedding in different coarsened networks. To our best knowledge, there are no other open-source defense baselines on graph matching available.

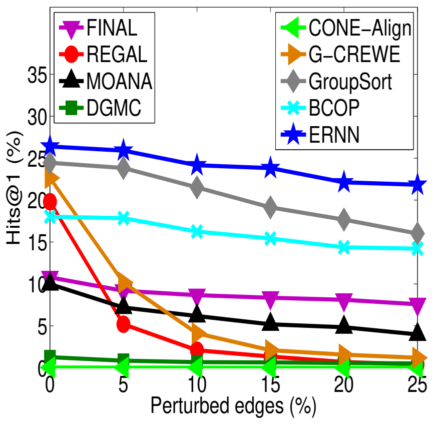
Attack models. We validate the robustness with four representative graph attack models. Random attack **(RND)** randomly adds and removes edges to generate perturbed graphs. **NEA** [9] is an efficient adversarial attack method that poison the network structure



(a) RND Attack

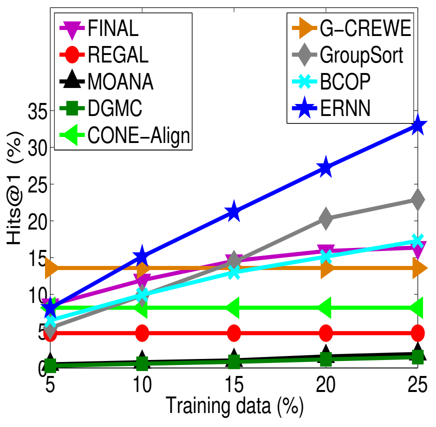


(b) NEA Attack

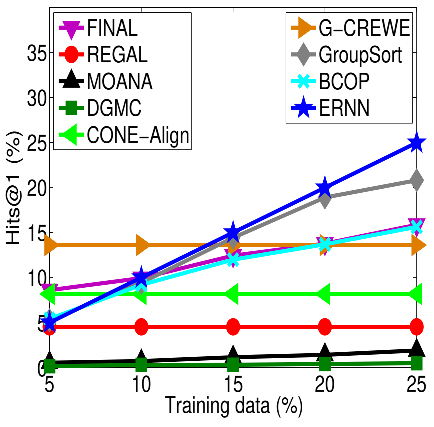


(c) GMA Attack

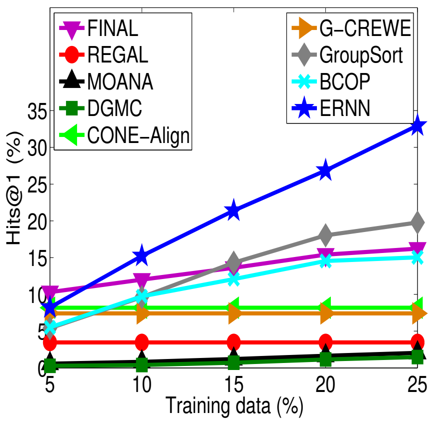
Figure 3.3: Matching on DBLP with varying perturbed edges



(a) RND Attack



(b) NEA Attack

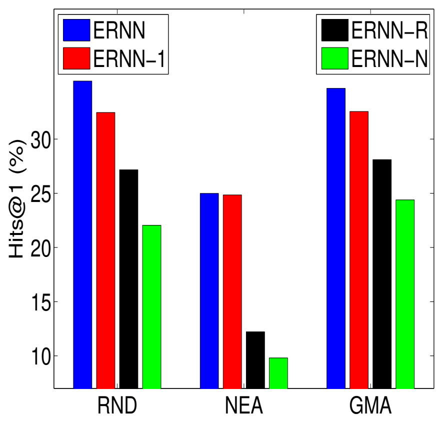


(c) GMA Attack

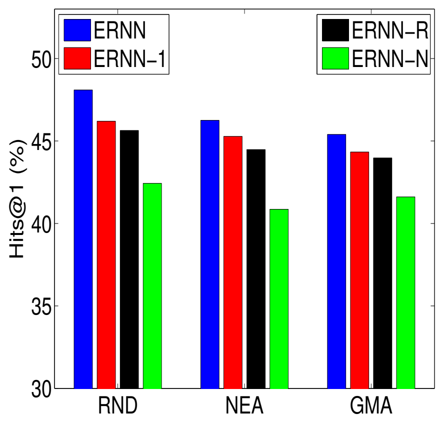
Figure 3.4: Matching on AS with varying training ratios

against both network embedding and node classification. **GMA** [102] is the only attack model on graph matching by pushing them to dense regions in two graphs to generate imperceptible and effective attacks. **RL-S2V** [18, 132] generates adversarial attacks on graph data based on reinforcement learning, which is used to attack both node classification and graph classification models.

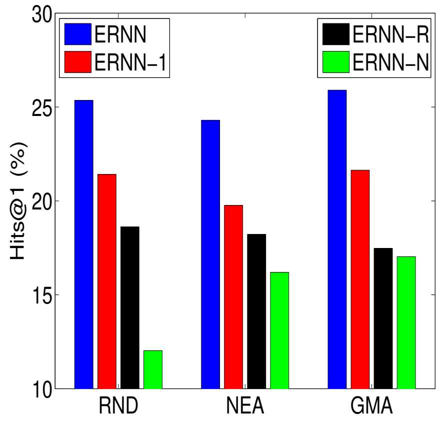
Variants of ERNN model. We evaluate four variants to show the strengths of different components. ERNN-1 utilizes a fixed $K_l = 1$ in our 1-Lipschitz neural network. ERNN-R



(a) AS

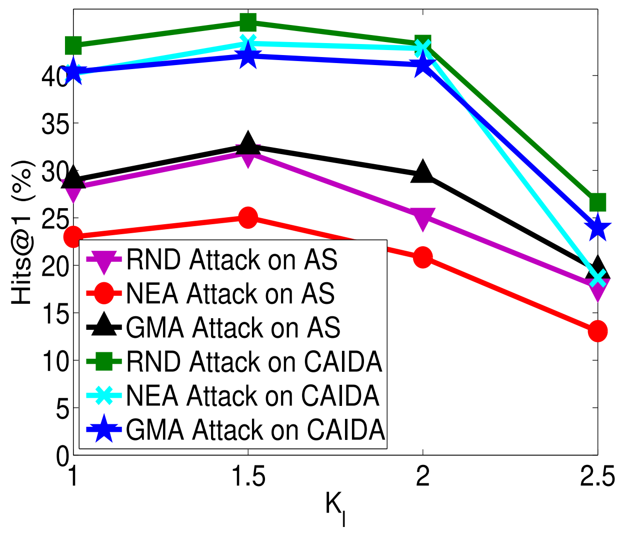


(b) CAIDA

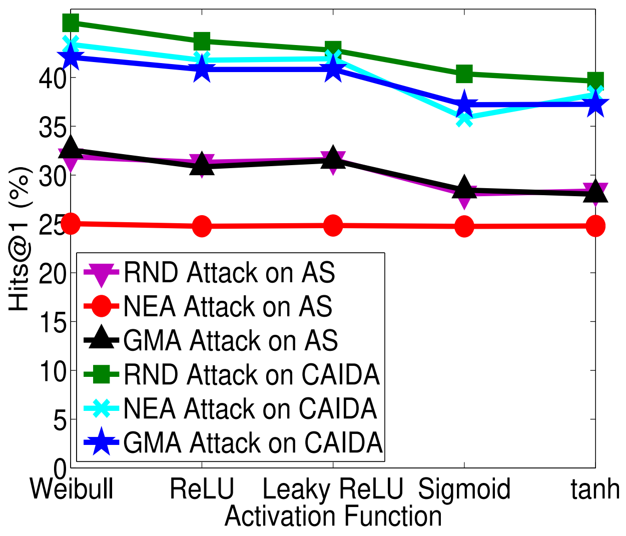


(c) DBLP

Figure 3.5: *Hits*@1 (%) of ERNN variants with 5% perturbed edges



(a) Fixed K_l



(b) Activation Function

Figure 3.6: *Hits*@1 (%) with varying parameters

employs the ReLU as the activation. ERNN-N only uses the regular fully-connected neural network. ERNN operates with the full support of graph-adaptive K_l, Weibull activation, and 1-Lipschitz neural network.

Evaluation metrics. We employ two measures to evaluation the quality of graph classification: *Accuracy* [42, 23, 40, 93] and root-mean-square error ($RMSE$). A higher *Accuracy* or a smaller $RMSE$ shows a better classification. In addition, we use $Hits@K$ [91, 27] to verify the quality of graph matching. A larger $Hits@K$ value indicates a better graph matching.

Defense performance on graph classification. Table 3.2 exhibits the *Accuracy* and $RMSE$ scores of nine graph classification algorithms under RL-S2V attacks over three groups of datasets. We randomly sample 30% of labeled graphs as training data and the rest as test data. The number of perturbed edges is fixed to 5% in these experiments. It is observed that among nine graph classification methods the ERNN method achieve the highest *Accuracy* and the smallest $RMSE$ on perturbed graphs in all experiments, showing the robustness of ERNN against adversarial attacks. Compared to the graph classification results by other models, ERNN, on average, achieves 15.2% *Accuracy* boost and 18.6% $RMSE$ improvement on three groups of datasets. In addition, the promising performance of ERNN over all three datasets implies that ERNN has great potential as a general robust graph classification solution to other datasets, which is desirable in practice.

Defense performance on graph matching with varying perturbation edges. Figures 3.1-3.3 present the graph matching quality under three attack models by varying the ratios of perturbed edges from 0% to 25%. We choose 30% of matched node pairs as training data. It is obvious that the quality by each matching algorithm decreases with increasing perturbed edges. This phenomenon indicates that current graph matching methods are sensitive to adversarial attacks. However, ERNN still achieves the highest $Hits@1$ values (> 0.249), which are better than other eight methods in most tests. Especially, when the perturbation ratio is larger than 10%, the $Hits@1$ drop by ERNN becomes slowly.

Table 3.3: Accuracy on image classification

Dataset	MNIST		CIFAR-10	
#Network Depth	**3**	**5**	**3**	**5**
GroupSort	0.91	0.91	0.48	0.50
BCOP	0.94	0.94	0.45	0.47
ERNN	**0.97**	**0.97**	**0.55**	**0.56**

Table 3.4: Wasserstein Distance estimation

Dataset	MNIST		CIFAR-10	
#Network Depth	**3**	**5**	**3**	**5**
GroupSort	2.31	2.55	2.23	2.74
BCOP	5.82	6.04	5.34	6.03
ERNN	**7.19**	**8.03**	**7.26**	**7.88**

Impact of training data ratios. Figure 3.4 shows the quality of nine graph matching algorithms on AS under three attack models by varying the ratio of training data from 5% to 25%. Here, the number of perturbed edges is fixed to 30%. We make the following observations on the performances by nine graph matching algorithms. (1) The performance curves keep increasing when the training data ratio increases. (2) ERNN outperforms other methods in most experiments with the highest $Hits@1$ scores: $> 5.01\%$. When there are appropriate training data available ($\geq 10\%$), the quality improvement by ERNN is obvious. A reasonable explanation is that more training data makes ERNN be more resilient to poisoning attacks under a small perturbation budget.

Ablation study. Figure 3.5 presents the $Hits@1$ scores of graph matching on three datasets with four variants of our ERNN model. We observe the complete ERNN achieves the highest $Hits@1$ ($> 24.9\%$) on AS, ($> 35.2\%$) over CAIDA, and ($> 17.5\%$) on DBLP, which are obviously better than other versions. Compared with ERNN-R, ERNN-1 performs well in most experiments. A reasonable explanation is that ReLU must trade nonlinear processing for gradient norm preservation, leading to less expressive neural networks. In addition, ERNN-R achieves the better performance than ERNN-N. A rational guess is that Lipschitz-bounded neural architecture is able to restrict the perturbation propagation on the neural networks, achieving remarkable robustness. These results illustrate all of graph-adaptive K_l, Weibull activation, and Lipschitz-bounded neural network are important in producing robust graph matching.

Table 3.5: Lipschitz constant on graph matching

Dataset	**AS**	**CAIDA**	**DBLP**
Unbounded Networks	2880	1530	1260
ERNN	**0.910**	**0.968**	**0.955**

Impact of K_l. Figure 4.3 (a) shows the impact of K_l in our ERNN model under three attack methods over three groups of datasets. The performance curves initially raise when K_l increases. As shown in the theoretical analysis, $K_l = 1$ is not the optimal solution and a large K_l can make the 1-Lipschitz neural network more robust. Later on, the performance curves keep relatively stable or even decreasing when K_l continuously increases. A reasonable explanation is that the too large K_l makes the norm-constrained weight matrices very small, such that it may hinder the feedforward of the neural network. Thus, it is important to choose the appropriate K_l for robust training.

Impact of activation function. Figure 4.3 (b) measures the effect of different activation functions in the ERNN model for the graph matching by using different activation functions. It is observed that among five activation functions, the $Hits@1$ values of our K_l-Lipschitz Weibull activation function outperforms all other competitors. This demonstrates that our Weibull activation is able to better maintain nonlinearity in Lipschitz-bounded neural networks. In addition, ReLU and Leaky ReLU achieve better performance than Sigmoid and tanh. This is consistent with the fact that non-saturating nonlinear activation functions often achieve faster training than saturating ones.

Validation of adversarial robustness on generic learning tasks. We conduct the experiments to validate the adversarial robustness of ERNN on two generic learning tasks: image classification and Wasserstein Distance estimation, by following similar setting in Table 2 in the GroupSort paper, as shown in Tables 3.3 and 3.4. We utilize two standard image datasets: MNIST [20] and CIFAR-10 [45]. Our ERNN model with Weibull activation still achieves the best performance in all tests.

Lipschitz constant estimate. The GroupSort and our ERNN models focus on designing 1-Lipschitz neural networks with enforcing each layer and entire network to be 1-Lipschitz. Table 3.5 shows the computed Lipschitz constants by ERNN and unbounded neural network on graph matching. The former is close to 1 and much smaller than the latter. This demonstrates that ERNN is able to successfully constrain the Lipschitz constant to 1.

3.5 Related Work

Lipschitz-bounded neural networks. Enforcing Lipschitz constraints in the training of neural networks is useful for ensuring adversarial robustness against adversarial attacks. Existing research activities can be classified into three broad categories: (1) Regularization techniques penalize or bound the Jacobian of the neural network, constraining the Lipschitz constant locally [72, 34, 44]. It is easy to train networks under the gradient penalties, but these methods cannot enforce the Lipschitz constraint globally; (2) Architecture constraint-based methods constrain the operator norm of each layer's weights, such as the matrix spectral norm [92, 61]. These approaches enforce the Lipschitz constraint but come at a cost in expressive power; and (3) Gradient norm preserving architectures enforce both weight norm and gradient norm as 1 to constraint 1-Lipschitz neural networks globally, which improves the expressive power to a certain degree [50, 17]. However, simply limiting the above two norms to 1 still sacrifices the expressive power, in comparison with regular neural networks without constrained weight and gradient.

Adversarial defenses on multiple graph learning. Graph data analysis have attracted active research in the last decade [12, 114, 115, 13, 109, 14, 47, 73, 116, 110, 63, 117, 111, 74, 119, 7, 121, 112, 118, 120, 48, 122, 108, 64, 124, 123, 69, 126, 125, 127, 113, 33, 84, 85, 128, 129, 102, 130, 131, 41, 86, 97]. The majority of existing techniques focus on tackling vulnerability and improving robustness on single graph learning tasks under adversarial attacks. Recently, researchers have demonstrated that multiple graph learning models, especially deep learning-based models, are highly sensitive to adversarial attacks, including graph classification [18, 76] and graph matching [102]. Several adversarial defense models have been developed to improve the robustness of multiple graph learning models in graph classification [98, 93, 40], graph matching [96], and multiple network embedding [129]. RGM is a robust graph matching model against visual noise, including image deformations, rotations, and outliers for image matching, but it fails to defend adversarial attacks on graph topology [96]. A common characteristics of the above techniques is that they often defend specific attacks on particular learning tasks, rather than attack-agnostic defense models.

3.6 Conclusions

In this work, we proposed an expressive 1-Lipschitz neural network to improve the robustness of multiple graph learning. First, the theoretical analysis is conducted to derive lower and upper bounds of feasible K_l under the 1-Lipschitz constraint. Second, a K_l-Lipschitz nonlinear activation function is designed to enforce the gradient norm as K_l at each layer. Finally, the nearest matrix orthogonalization and polar decomposition techniques are utilized to constraint the weight norm as $1/K_l$.

Chapter 4
Unsupervised Federated Graph Matching with Graphlet Extraction and Separate Trust Region

4.1 Introduction

Federated graph learning (FGL) is a promising paradigm that enables collaborative training of shared machine learning models over large-scale distributed graph data, while preserving privacy of local data [306, 223, 298]. Only recently, researchers have started to attempt to study the FGL problems [277, 256, 308, 240, 280, 223, 246, 289, 283, 234, 235]. Graph matching (i.e., network alignment) is one of the most important research topics in the graph domain, which aims to match the same entities (i.e., nodes) across two or more graphs [165, 167, 188, 172, 213, 15, 191]. It has been widely applied to many real-world applications ranging from protein network matching in bioinformatics [154, 180], user account linking in different social networks [209, 157, 105, 144, 158], and knowledge translation in multilingual knowledge bases [199, 214], to geometric keypoint matching in computer vision [27]. While the existing techniques have achieved remarkable performance in the above graph learning domains, there is still a paucity of techniques of effective federated graph matching (FGM), which is much more difficult to study. Directly sharing and inferring matched node pairs on different graphs across clients and local graphs over multiple clients gives rise to a serious privacy leakage concern. In this work, we aim to answer the following questions: (1) How to train effective FGM models on distributed clients with maintaining high matching performance? (2) How to make FGM models with strong privacy protection for cross-client information exchange?

Research activities on centralized graph matching can be classified into two broad categories: supervised graph matching [105, 91, 49, 15, 27] and unsupervised graph matching [213, 35, 157, 38]. The former utilizes a set of pre-matched node pairs between pairwise graphs belonging to the same entities as training data to learn an effective graph matching model by maximizing the similarities (or minimizing the distances) between the pre-matched node pairs. The node pairs with the largest similarities between pairwise graphs in test data are identified as the one-to-one matching results. The latter fails to employ the strength of training data and thus often leads to sub-optimal solutions. Unfortunately, supervised graph matching methods that use the pre-matched node pairs as the training data is improper for the FGM scenarios due to privacy risks of direct cross-client information exchange when the graph data are distributed over different clients.

This motivates us to capture nodes' graphlet features to generate pseudo matched node pairs on different graphs across clients as the pseudo training data for leveraging the strength of supervised graph matching. A graphlet is a small graph of size up to k nodes of a larger graph, such as triangle, wedge, or k-clique, which describes the local topology of a larger graph [274, 275, 242, 261]. A node can be described by a graphlet feature vector, where each component denotes the frequency of one type of graphlets. It is highly possible that the nodes in different graphs with the large similarities regarding their graphlet features correspond to the same entities. Thus, they can be treated as the pseudo matched node pairs for pseudo supervised FGM.

However, graphlet enumeration one by one on large-scale graphs is impossible due to expensive cost. We propose to leverage Monte Carlo Markov Chain (MCMC) technique for sampling a small number of graphlets. The number of graphlet samples is much smaller than that of all graphlets in the graphs, which dramatically improves the efficiency of graphlet enumeration. Theoretical analysis is conducted to demonstrate that the estimated graphlet count based on the MCMC sampling strategy is close to the actual count of all graphlets, which implies that the graphlet samples and all graphlets share similar distributions.

In order to maintain the privacy requirement of federated learning, we first encrypt local graph data on each client. The encrypted graph data from all clients are uploaded to the server for matching the graphs with each other. Note that stochastic gradient descent (SGD) optimization widely used in deep learning fails to work on the clients in the FGM, since each client can access only its own local graph data and thus cannot update local loss based on the pseudo matched node pairs. We propose a separate trust region algorithm for pseudo supervised FGM while maintaining the privacy constraints. Specifically, we separate model optimization from model evaluation in the trust region algorithm: (1) the server aggregates the local model parameter M_b^s on each client s into a global model parameter M_b at global iteration b, runs and evaluates M_b on the all pseudo training data $\tilde{D}^{st}$ and the encrypted graph data, and computes the individual loss $\mathcal{L}^s(M_b)$, the gradient $\nabla\mathcal{L}^s(M_b)$, and the Hessian $\nabla^2\mathcal{L}^s(M_b)$ for each client s; (2) client s receives its individual $\mathcal{L}^s(M_b)$, $\nabla\mathcal{L}^s(M_b)$, and $\nabla^2\mathcal{L}^s(M_b)$ from the server and optimizes M_{b+1}^s.

Unfortunately, the second-order Hessian computation $\nabla^2\mathcal{L}^s(M_b)$ in the separate trust region algorithm is time-consuming over large-scale graph data. We propose to explore quasi-Newton conditions to construct a positive definite scalar matrix $\alpha_b\mathbf{I}$, where $\alpha_b \geq 0$ is a scalar and $\mathbf{I}$ is an identify matrix, as the Hessian approximation with only first-order gradients, i.e., $z^T\nabla^2\mathcal{L}^s(M_b)z \approx \alpha_b z^T z$. We theoretically derive the error introduced by the separate trust region due to the Hessian approximation and conduct the convergence analysis of the approximation method.

To our best knowledge, this work is the first to offer an unsupervised federated graph matching solution for inferring matched node pairs on different graphs across clients while maintaining the privacy requirement of federated learning, by leveraging the graphlet theory and trust region optimization. Our UFGM method exhibits three compelling advantages: (1) The combination of the unsupervised FGM and the encryption of local raw graph data is able to provide strong privacy protection for sensitive local data; (2) The graphlet feature extraction can leverage the strength of supervised graph matching with the pseudo training data for improving the matching quality; and (3) The separate trust region for pseudo supervised FGM is helpful to enhance the efficiency while maintaining the privacy constraints.

Empirical evaluation on real datasets demonstrates the superior performance of our UFGM model against several state-of-the-art centralized graph matching, federated domain adaption, and FGL methods.

4.2 Background

4.2.1 Supervised Graph Matching

Given a set of S graphs $G = \{G^1, \cdots, G^S\}$. Each graph is denoted as $G^s = (V^s, E^s)$ $(1 \leq s \leq S)$, where $V^s = \{v_1^s, v_2^s, \cdots\}$ is the set of nodes and $E^s = \{(v_i^s, v_j^s) : 1 \leq i, j \leq |V^s|, i \neq j\}$ is the set of edges. Each G^s has a binary adjacency matrix $\mathbf{A}^s$, where each entry $\mathbf{A}_{ij}^s = 1$ if there exists an edge $(v_i^s, v_j^s) \in E^s$; otherwise $\mathbf{A}_{ij}^s = 0$. $\mathbf{A}_{i:}^s$ specifies the i^{th} row vector of $\mathbf{A}^s$ and is used to denote the representation of a node v_i^s.

The entire training data consist of a set of training data between pairwise graphs, i.e., $D = \{D^{12}, \cdots, D^{1S}, \cdots, D^{(S-1)S}\}$. Each D^{st} $(1 \leq s < t \leq S)$ specifies a set of pre-matched node pairs $D^{st} = \{(v_i^s, v_j^t) | v_i^s \leftrightarrow v_j^t, v_i^s \in V^s, v_j^t \in V^t\}$, where $v_i^s \leftrightarrow v_j^t$ represents that two nodes v_i^s and v_j^t are the equivalent ones in two graphs G^s and G^t and are treated as the same entity. The objective of supervised graph matching is to utilize D^{st} as the training data to identify the one-to-one matchings between nodes v_i^s and v_j^t in the test data. Based on structure, attribute, or embedding features, existing efforts often aim to learn an matching function M to map the node pairs $(v_i^s, v_j^t) \in D^{st}$ with different features across two graphs into common space, i.e, minimize the distances between source nodes $M(v_i^s)$ and target ones $M(v_j^t)$ [59, 105, 91, 158]. The node pairs $(v_i^s, v_j^t) \in D^{st}$ with the smallest distances in the test data are selected as the matching results.

$$\mathcal{L} = \sum_{s=1}^{S} \sum_{t=s+1}^{S} \mathbb{E}_{(v_i^s, v_j^t) \in D^{st}} \|M(v_i^s) - M(v_j^t)\|_2^2 \tag{4.1}$$

Graph convolutional networks (GCNs) have demonstrated their superior learning performance in network embedding tasks [247]. In this paper, if there are no specific descriptions, we utilize the GCNs to learn the embedding representation of each node v_i^s in each graph G^s, based on its original structure features $\mathbf{A}_{i:}^s$. The embedding representation of v_i^s is denoted by $\mathbf{v}_i^s$. Thus, the objective of supervised graph matching is reformulated as follows.

$$\mathcal{L} = \sum_{s=1}^{S} \sum_{t=s+1}^{S} \mathbb{E}_{(v_i^s, v_j^t) \in D^{st}} \| M(\mathbf{v}_i^s) - M(\mathbf{v}_j^t) \|_2^2 \tag{4.2}$$

4.2.2 Federated Graph Matching

In this paper, without loss of generality, we assume that each client contains only one local graph in the federated setting, but it is straightforward to extend to the case of multiple local graphs owned by each client. Given S clients with a set of S graphs $\mathcal{G} = \{G^1, \cdots, G^S\}$ and their local training data $D = \{D^{12}, \cdots, D^{1S}, \cdots, D^{(S-1)S}\}$, and a server, federated graph matching (FGM) aims to learn a global graph matching model M on the server by optimizing the problem below.

$$\begin{aligned} \min_{M \in \mathbb{R}^d} \mathcal{L}(M) &= \sum_{s=1}^{S} \mathcal{L}^s(M) = \sum_{s=1}^{S} \sum_{t=s+1}^{S} \frac{N^{st}}{N} L^{st}(M) \\ \text{where } L^{st}(M) &= \frac{1}{N^{st}} \sum_{(v_i^s, v_j^t) \in D^{st}} l_{ij}^{st}(M) \end{aligned} \tag{4.3}$$

where $l_{ij}^{st}(M) = \|M(\mathbf{v}_i^s) - M(\mathbf{v}_j^t)\|_2^2$ denotes the loss function of the prediction on the pre-matched node pair $(v_i^s, v_j^t) \in D^{st}$ made with M. $\mathcal{L}^s(M)$ and $\mathcal{L}(M)$ are the local loss function on client s and the global one respectively. $N^{st} = |D^{st}|$ denotes the size of local training dataset D^{st}. N is the size of total training data D, i.e., $N = N^{12} + \cdots + N^{1S} + \cdots + N^{(S-1)S}$. A local graph matching model M^s is optimized based on the local loss $\mathcal{L}^s(M)$. In the FGM, M is iteratively updated with the aggregation of all $M^1, \cdot, M^S$ on S clients in each round, i.e., $M = \sum_{s=1}^{S} \sum_{t=s+1}^{S} \frac{N^{st}}{N} M^s$.

Observed from Eq.(3), when calculating the local loss $\mathcal{L}^s(M)$ on client s for optimizing the local model M^s, we need to access the pre-matched node pairs $\{v_i^s, v_j^t\} \in D^{st}$ and the graph G^t on client t. This operation obviously violates the privacy requirement of federated learning. Thus, it is difficult to utilize the pre-matched node pairs for supervised FGM.

4.3 Monte Carlo Markov Chain for Graphlet Feature Extraction

As discussed in the last section, the supervised graph matching usually achieves better performance than the unsupervised one. In addition, supervised FGM may lead to serious privacy concerns. In this work, we explore to capture nodes' graphlet features to generate pseudo matched node pairs on different graphs across clients as the pseudo training data for leveraging the strength of supervised graph matching while keeping the local graph data safe.

In order to prohibit other clients and server from accessing local raw graphs and embedding representations on any client s for maintaining the privacy requirement of FGM, we first utilize an efficient matrix generation method [270] to produce a random nonsingular matrix $\mathbf{K}$ as a key. Each client employs $\mathbf{K}$ to encrypt its network embedding $\hat{\mathbf{v}}_i^s = \mathbf{v}_i^s \mathbf{K}$ from the original one $\mathbf{v}_i^s$ and uses its inverse $\mathbf{K}^{-1}$ to decrypt from $\hat{\mathbf{v}}_i^s$ to $\mathbf{v}_i^s = \hat{\mathbf{v}}_i^s \mathbf{K}^{-1}$. The encrypted $\hat{\mathbf{v}}_i^s$ from all clients will be uploaded to the server for graph matching. It is important that $\mathbf{K}$ is kept secret between senders and recipients. In our setting, $\mathbf{K}$ is shared by all clients, but not accessed by the server.

The first step of graphlet feature extraction is to enumerate all graphlets in a graph $G = (V, E)$. Concretely, let G_k be the set of all C connected induced k-subgraphs (with k nodes) in G. Let $\mathcal{G}\dashv_1, \mathcal{G}_2, \cdots, \mathcal{G}_R$ be all R types of non-isomorphic k-graphlets (with k nodes) for which we would like to count. We denote a k-subgraph $g \in G_k$ that is isomorphic to a k-graphlet $\mathcal{G}_r$ $(1 \leq r \leq R)$ as $g \sim \mathcal{G}_r$. The number of k-graphlets of type r in G is equal to

$$n_{kr}(G) = \sum_{g \in G_k} \mathbb{I}\left(g \sim \mathcal{G}_r\right) \tag{4.4}$$

where $\mathbb{I}(\cdot)$ is an indicator function.

However, graphlet enumeration one by one on large-scale graphs is impossible due to expensive cost. We propose a MCMC sampling technique for which one can calculate the stationary distribution p on the k-subgraphs in G_k. We only sample a small number of k-subgraphs $g_{k1}, \cdots, g_{kO}$ in G, where the size $O << C$. Then we use Horvitz-Thompson inverse probability weighting to estimate the graphlet counts as follows.

$$\tilde{n}_{kr}(G) = \frac{1}{O} \sum_{o=1}^{O} \frac{\mathbb{I}\left(g_{ko} \sim \mathcal{G}_r\right)}{p(g_{ko})} \tag{4.5}$$

Next, we describe how to expand from 1-subgraphs to k subgraphs in the graphlet enumeration. For any $(k-1)$-subgraph g_{k-1}, we expend it to a k-subgraph by adding a node from its neighborhood $\mathcal{N}_v(g_{k-1})$ at random in terms of a certain probability distribution, where $\mathcal{N}_v(g_{k-1})$ is the set of all nodes adjacent to a certain node in g_{k-1} but not including all nodes in g_{k-1}.

This expansion operation can explore any subgraph in G_k. It iteratively builds a k-subgraph g_k from a starting node. First, suppose that a starting node v_1 is sampled from the distribution q, which can be computed from local information. We assume that $q(v) = \frac{f(\deg(v))}{F}$, where $f(x)$ is a certain function (usually a polynomial) and F is a user-defined normalizing factor. Thus, a 1-subgraph $g_1 = \{v_1\}$ is generated. Second, it samples an edge (v_1, v_2) uniformly in $\mathcal{N}_e(g_1)$, where $\mathcal{N}_e(g_1)$ is the set of all edges that connect a node in g_1 and a node outside of g_1. Thus, a node v_2 is then attached to g_1, forming a 2-subgraph $g_2 = g_1 \cup v_2 \cup (v_1, v_2)$. Similarly, at each iteration, it samples an edge (v_i, v_{j+1}) $(1 \leq i \leq j)$ from $\mathcal{N}_e(g_j)$ uniformly at random and attach the node v_{j+1} to the subgraph g_j, forming a $j+1$-subgraph $g_{j+1} = g_j \cup v_{j+1} \cup (v_i, v_{j+1})$. After $k-1$ iterations, we obtain a k-subgraph

g_k. Once g_k has been sampled we need to classify it into a graphlet type, i.e., $g_k \sim \mathcal{G}_r$. The method repeats the above process O times until O k-graphlets $g_{k1}, g_{k1}, \cdots, g_{kO}$ are produced.

We conduct the theoretical analysis to evaluate the permanence of our graphlet enumeration based on the MCMC sampling, in terms of the difference between the estimated and actual graphlet counts.

In the estimation $\tilde{n}_{kr}(G)$ in Eq.(5), a key problem is to calculate $p(g_{ko})$. The probability $p(g_k)$ of getting a k-subgraph g_k via subgraph expansion from a $(k-1)$-subgraph g_{k-1} is given by the sum $p(g_k) = \sum_{g_{k-1}} \mathbb{P}(g_k|g_{k-1})p(g_{k-1})$, where the sum is taken over all connected $(k-1)$-subgraphs $g_{k-1} \subset g_k$, and $\mathbb{P}(g_k|g_{k-1})$ is the probability of getting from g_{k-1} to g_k in the expansion process.

$$p(g_k) = \sum_{g_{k-1} \subset g_k} p(g_{k-1}) \frac{\deg_{g_{k-1}}\left(V_{g_k} - V_{g_{k-1}}\right)}{|\mathcal{N}_e(g_{k-1})|} = \sum_{g_{k-1} \subset g_k} p(g_{k-1}) \frac{|E_{g_k}| - \left|E_{g_{k-1}}\right|}{\sum_{v \in V_{g_{k-1}}} \deg(v) - 2\left|E_{g_{k-1}}\right|} \tag{4.6}$$

where for a subgraph $g_k \subseteq G$, V_{g_k} the set of its nodes and E_{g_k} is the set of its edges. $\deg_{g_{k-1}}(V)$ specifies the number of nodes in g_{k-1} that are connected to a node set V. $\deg(v)$ denotes the number of associated edges of a node v.

In order to calculate $p(g_k)$, we need to consider all possible orderings of nodes in g_k. Assume that the original node ordering of g_k via the subgraph expansion is $x_k = \{v_1, v_2, \cdots, v_k\}$. Let $\mathcal{S}(g_k) = [v_1, v_2, \cdots, v_k]$ be the set of all possible node sequences of x_k. Notice that an induced subgraph $h_l(x_k) = \{v_1, v_2, \cdots, v_l, x_k, G\}$ of graph G with the first l nodes $\{v_1, v_2, \cdots, v_l\}$ in x_k must be a connected subgraph for any l ($1 \leq l \leq k$). Thus, we have

$$\mathcal{S}(g_k) = \{[v_1, \ldots, v_k] | \{v_1, \ldots, v_k\} = V_{g_k}, g_k | \{v_1, \ldots, v_l\} \text{is connected}\} \tag{4.7}$$

The following theorems give an explicit solution of the probability $p(g_k)$ of getting a k-subgraph g_k via subgraph expansion and the variance of the estimation $\tilde{n}_{kr}(G)$ of graphlet counts.

Theorem 4.1 *Let $x_k = \{v_1, v_2, \cdots, v_k\}$ be the original node ordering of g_k via the subgraph expansion, $\mathcal{S}(g_k) = [v_1, v_2, \cdots, v_k]$ be the set of all possible node sequences of x_k, $x_k[i]$ be the i^{th} node in x_k, F be a user-defined normalizing factor in the subgraph expansion, and $h_l(x_k) = \{v_1, v_2, \cdots, v_l, x_k, G\}$ be an induced subgraph of graph G with the first l nodes $\{v_1, v_2, \cdots, v_l\}$ in x_k, then the probability of getting a k-subgraph g_k via the subgraph expansion is*

$$p(g_k) = \sum_{x_k \in \mathcal{S}(g_k)} \frac{f(\deg(x_k[1]))}{F} \prod_{l=1}^{k-1} \frac{\left|E_{h_{l+1}(x_k)}\right| - \left|E_{h_l(x_k)}\right|}{\sum_{i=1}^{l} \deg(x_k[i]) - 2\left|E_{h_l(x_k)}\right|} \tag{4.8}$$

Theorem 4.2 *Let $\tilde{n}_{kr}(G) = \frac{1}{O}\sum_{o=1}^{O} \frac{\mathbb{I}(g_{ko} \sim \mathcal{G}_r)}{p(g_{ko})}$ be the estimation of graphlet counts, $d_1, \cdots, d_k$ be the k highest degrees of nodes in G, and denote $D = \prod_{l=2}^{k-1}(d_1 + \cdots + d_k)$. If q for sampling the starting node is the stationary distribution of the node random walk, then the upper bound of the variance* $\mathrm{Var}(\tilde{n}_{kr}(G))$ *is*

$$\mathrm{Var}(\tilde{n}_{kr}(G)) \leq \frac{1}{O} n_{kr}(G) \frac{2\left|E_G\right|}{\left|\mathcal{S}(\mathcal{G}_r)\right|} D \tag{4.9}$$

It is observed that the variance $\mathrm{Var}(\tilde{n}_{kr}(G))$ is small when the distribution of $p(g_k)$ is close to uniform distribution. A larger $p(g_k)$ results in a smaller variance of the estimator. Thus, the variation can be reduced by an appropriate choice of q for sampling the starting node, say a smaller normalizing factor F. In this case, the estimated graphlet count $\tilde{n}_{kr}(G)$ is close to the actual count $n_{kr}(G)$, which implies that the graphlet samples and all graphlets share similar distributions.

We capture the graphlet features of a node by computing the frequency of each type of graphlet with size up to k that is associated with this node. For the node pairs between pairwise graphs, we compute the cosine similarity scores based on the graphlet features on all R types of graphlet. The top-K node pairs with the largest similarities between pairwise graphs G^s and G^t are treated as the pseudo matched node pairs and added to the pseudo training data $\tilde{D}^{st}$.

4.4 Separate Trust Region for Unsupervised Federated Graph Matching

In this work, according to the graphlet-based pseudo training data $\tilde{D}^{st}$ and the encrypted network embedding $\hat{\mathbf{v}}_i^s$, we propose a separate trust region algorithm for pseudo supervised FGM while maintaining the privacy constraints. Specifically, we separate model optimization from model evaluation in the trust region algorithm: (1) the server aggregates the local model parameter M_b^s on each client s into a global model parameter M_b at global iteration b, runs and evaluates M_b on the all $\tilde{D}^{st}$ and $\hat{\mathbf{v}}_i^s$, and computes the individual loss $\mathcal{L}^s(M_b)$, the gradient $\nabla\mathcal{L}^s(M_b)$, and the Hessian $\nabla^2\mathcal{L}^s(M_b)$ for each client s; (2) client s receives its individual $\mathcal{L}^s(M_b)$, $\nabla\mathcal{L}^s(M_b)$, and $\nabla^2\mathcal{L}^s(M_b)$ from the server and optimizes M_{b+1}^s.

$$\begin{gathered}\textbf{Server}: \text{Compute}\, M_b = \sum_{s=1}^{S}\sum_{t=s+1}^{S}\frac{N^{st}}{N}M_b^s,\ L^{st}(M_b) = \frac{1}{N^{st}}\sum_{(v_i^s,v_j^t)\in\tilde{D}^{st}}\|M_b(\hat{\mathbf{v}}_i^s) - M_b(\hat{\mathbf{v}}_j^t)\|_2^2,\\ \mathcal{L}^s(M_b) = \sum_{t=s+1}^{S}\frac{N^{st}}{N}L^{st}(M_b),\ \nabla\mathcal{L}^s(M_b),\ \text{and}\, \nabla^2\mathcal{L}^s(M_b)\end{gathered} \tag{4.10}$$

$$\begin{aligned}&\textbf{Client s}: \text{Optimize}\, z^* = \arg\min u_b(z) = \mathcal{L}^s(M_b) + (\nabla\mathcal{L}^s(M_b))^T z + \frac{1}{2}z^T\nabla^2\mathcal{L}^s(M_b)z,\ \text{s.t.}\|z\| \leq \Delta^s\\ &\qquad\qquad \text{Update}\, M_{b+1}^s = M_b^s + z^*\end{aligned} \tag{4.11}$$

where $\Delta^s > 0$ is the trust-region radius. z^* is the trust-region step. The individual loss $\mathcal{L}^s(M_b)$ aims to minimize the sum of distance between nodes on client s and nodes on other clients in the pseudo training data $\tilde{D}^{st}$. The node pairs with the smallest distance between pairwise encrypted network embeddings are selected as the matching results.

A key challenge in the separate trust region algorithm is to compute the second-order Hessian computation $\nabla^2 \mathcal{L}^s(M_b)$. It is time-consuming over large-scale graph data. We propose to explore quasi-Newton conditions to construct a positive definite scalar matrix $\alpha_b \mathbf{I}$, where $\alpha_b \geq 0$ is a scalar and $\mathbf{I}$ is an identify matrix, as the Hessian approximation with only first-order gradients, i.e., $z^T \nabla^2 \mathcal{L}^s(M_b) z \approx \alpha_b z^T z$.

Concretely, the quasi-Newton condition is given as follows.

$$\nabla^2 \mathcal{L}^s(M_b) z_b = y_b \tag{4.12}$$

where $z_b = M_{b+1} - M_b$ and $y_b = \nabla \mathcal{L}^s(M_{b+1}) - \nabla \mathcal{L}^s(M_b)$. The condition is derived from the following quadratic model.

$$u_{b+1}(z) = \mathcal{L}^s(M_{b+1}) + (\nabla \mathcal{L}^s(M_{b+1}))^T z + \frac{1}{2} z^T \nabla^2 \mathcal{L}^s(M_{b+1}) z \tag{4.13}$$

The quadratic model is an approximation of the objective function at iteration $b+1$ and satisfies the following three interpolation conditions:

$$(1)\ u_{b+1}(0) = \mathcal{L}^s(M_{b+1}), \quad (2)\ \nabla u_{b+1}(0) = \nabla \mathcal{L}^s(M_{b+1}), \quad (3)\ \nabla u_{b+1}(-z_b) = \nabla \mathcal{L}^s(M_b) \tag{4.14}$$

It is difficult to satisfy the quasi-Newton equation in Eq.(12) with a nonsingular scalar matrix [229]. A recent study introduced a weak condition form by projecting the quasi-Newton equation in Eq.(12) in the direction z_b [238].

$$z_b^T \nabla^2 \mathcal{L}^s(M_{b+1}) z_b = z_b^T y_b \tag{4.15}$$

The choice of z_b may influence the quality of the curvature information provided by the weak quasi-Newton condition. Another weak condition is directly derived from an interpolation emphasizing more on function values rather than from the projection of the quasi-Newton condition [290].

$$u_{b+1}(-z_b) = \mathcal{L}^s(M_b) \tag{4.16}$$

By combining sub-conditions (1) and (2) in Eq.(14) and replacing (3) with Eq.(16), we can get another weak quasi-Newton condition.

$$z_b^T \nabla^2 \mathcal{L}^s(M_{b+1}) z_b = 2\left(\mathcal{L}^s(M_b) - \mathcal{L}^s(M_{b+1}) + z_b^T \nabla \mathcal{L}^s(M_{b+1})\right) \tag{4.17}$$

By integrating two types of weak quasi-Newton conditions together, we have a generalized weak quasi-Newton condition.

$$\begin{aligned} z_b^T \nabla^2 \mathcal{L}^s(M_{b+1}) z_b &= (1-\omega) z_b^T y_b + \omega \left[2\left(\mathcal{L}^s(M_b) - \mathcal{L}^s(M_{b+1})\right) + 2 z_b^T \nabla \mathcal{L}^s(M_{b+1})\right] \\ &= z_b^T y_b + \omega \left[2\left(\mathcal{L}^s(M_b) - \mathcal{L}^s(M_{b+1})\right) + \left(\nabla \mathcal{L}^s(M_b) + \nabla \mathcal{L}^s(M_{b+1})\right)^T z_b\right] \end{aligned} \tag{4.18}$$

where $\omega \geq 0$ is the weight. If $\nabla^2 \mathcal{L}^s(M_{b+1})$ is set to be a scalar matrix $\alpha^*_{b+1}(\omega)\mathbf{I}$, then we have

$$\alpha_{b+1}(\omega) = \frac{z_b^T y_b + \omega \left[2\left(\mathcal{L}^s(M_b) - \mathcal{L}^s(M_{b+1})\right) + \left(\nabla \mathcal{L}^s(M_b) + \nabla \mathcal{L}^s(M_{b+1})\right)^T z_b\right]}{z_b^T z_b} \tag{4.19}$$

The following theorems derive the error introduced by the separate trust region due to the Hessian approximation and conduct the convergence analysis of the approximation method.

Theorem 4.3 *Let d be the dimension of the flattened M_{b+1}, $\otimes$ be an appropriate tensor product, $\mathcal{A}_{b+1} \in \mathbb{R}^{d\times d\times d}$ and $\mathcal{B}_{b+1} \in \mathbb{R}^{d\times d\times d\times d}$ are the tensors of $\mathcal{L}^s(M_{b+1})$ at iteration $b+1$ satisfying*

$$\mathcal{A}_{b+1} \otimes z_b^3 = \sum_{i,j,k=1}^{d} \frac{\partial^3 \mathcal{L}^s(M_{b+1})}{\partial M^i \partial M^j \partial M^k} z_b^i z_b^j z_b^k \tag{4.20}$$

and

$$\mathcal{B}_{b+1} \otimes z_b^4 = \sum_{i,j,k,l=1}^{d} \frac{\partial^4 \mathcal{L}^s(M_{b+1})}{\partial M^i \partial M^j \partial M^k \partial M^l} z_b^i z_b^j z_b^k z_b^l. \tag{4.21}$$

Suppose that $\mathcal{L}^s(M_{b+1})$ is sufficiently smooth, if $||z_b||$ is small enough, then we have

$$z_b^T \nabla^2 \mathcal{L}^s(M_{b+1}) z_b - \alpha_{b+1}(\omega) z_b^T z_b = \left(\frac{1}{2} - \frac{\omega}{6}\right) \mathcal{A}_{b+1} \otimes z_b^3 - \left(\frac{1}{6} - \frac{\omega}{12}\right) \mathcal{B}_{b+1} \otimes z_b^4 + \mathcal{O}\left(\|z_b\|^5\right) \tag{4.22}$$

Theorem 4.4 *Suppose $\|\nabla \mathcal{L}^s(M_b)\| \neq 0$, the solution z_b of the separate trust region optimization $\arg\min u_b(z) = \mathcal{L}^s(M_b) + (\nabla \mathcal{L}^s(M_b))^T z + \frac{1}{2} z^T \nabla^2 \mathcal{L}^s(M_b) z,\ s.t.\|z\| \leq \Delta^s$ in Eq.(**??**) satisfies*

$$u_b(0) - u_b(z_b) \geq \frac{1}{2} \|\nabla \mathcal{L}^s(M_b)\| \min\left\{\Delta^s, \frac{\|\nabla \mathcal{L}^s(M_b)\|}{\alpha_b}\right\} \tag{4.23}$$

Finally, the separate trust region optimization based on two weak quasi-Newton conditions is given below.

$$z^* = \arg\min u_b(z) \approx \mathcal{L}^s(M_b) + (\nabla \mathcal{L}^s(M_b))^T z + \frac{1}{2} \alpha_b(\omega) z^T z,\ \text{s.t.}\|z\| \leq \Delta^s \tag{4.24}$$

Table 4.1: Statistics of the Datasets

Dataset	#Clients/#Graphs	#Avg. Nodes	#Nodes	#Avg. Edges	#Edges
SNS	3	14,331	14,262 $\sim$ 14,573	51,358	48,105 $\sim$ 53,381
PPI	50	1,767	1,767	32,320	31,179 $\sim$ 32,358
DBLP	20	10,038	9,984 $\sim$ 10,168	56,314	54,891 $\sim$ 60,058

4.5 Experimental Evaluation

In this section, we have evaluated the performance of our UFGM model and other comparison methods for federated graph matching over serval representative federated graph datasets to date. We show that UFGM with graphlet feature extraction and separate trust region is able to achieve higher matching accuracy and faster convergence in federated settings against several state-of-the-art centralized graph matching, federated graph learning and federated domain adaption methods.

Datasets. We focus on three popular computer vision and natural language processing tasks over three representative benchmark datasets respectively: social networks (SNS) [207], protein-protein interaction networks (PPI) [313], and DBLP coauthor graphs (DBLP) [3], as shown in Table 4.1. Without loss of generality, we assume that each client contains only one local graph in the federated setting. For the supervised learning methods, the training data ratio over the above three datasets is all fixed to 20%. We train the models on the training set and test them on the test set for three datasets.

Baselines. To our best knowledge, this work is the first to offer an unsupervised federated graph matching solution for inferring matched node pairs on different graphs across clients while maintaining the privacy requirement of federated learning, by leveraging the graphlet theory and trust region optimization. Thus, we choose three types of baselines that are most close to the task of federated graph matching: centralized graph matching, federated graph learning and federated domain adaption. We compare the UFGM model with six state-of-the-art centralized graph matching models: **NextAlign** [303], **NetTrans** [302],

Table 4.2: Final Performance on SNS

Type	Algorithm	*Hits*@1	*Hits*@5	*Hits*@10	*Hits*@50	*Loss*
Centralized Graph Matching	NextAlign	**0.951**	**0.962**	**0.972**	**0.979**	2.115
	NetTrans	0.921	0.932	0.958	0.960	1.571
	CPUGA	0.248	0.392	0.433	0.563	2.598
	ASAR-GM	0.299	0.394	0.453	0.668	1.699
	SIGMA	0.499	0.560	0.633	0.782	1.652
	SeedGNN	0.884	0.943	0.959	0.960	3.039
Federated Domain Adaption	DualAdapt	0.006	0.006	0.007	0.011	2.106
	EFDA	0.007	0.011	0.014	0.029	3.249
	WSDA	0.009	0.011	0.013	0.016	2.746
	FKA	0.005	0.006	0.006	0.008	2.227
Federated Graph Learning	FedGraphNN	0.081	0.132	0.179	0.200	4.259
	FKGE	0.231	0.323	0.352	0.441	0.817
	SpreadGNN	0.115	0.179	0.213	0.236	**0.290**
	SFL	0.000	0.001	0.001	0.002	5.285
	FederatedScope-GNN	0.001	0.001	0.001	0.002	4.259
	FedStar	0.057	0.092	0.137	0.211	2.255
	UFGM	0.771	0.880	0.902	0.930	0.659

CPUGA [263], **ASAR-GM** [272], **SeedGNN** [296], and **SIGMA** [249], six representative federated graph learning architectures: **FedGraphNN** [233], **FKGE** [264], **SpreadGNN** [236], **SFL** [225], **FederatedScope-GNN** [287], and **FedStar** [278], and four recent federated domain adaption methods: **DualAdapt** [265], **EFDA** [243], **WSDA** [239], and **FedKA** [276].

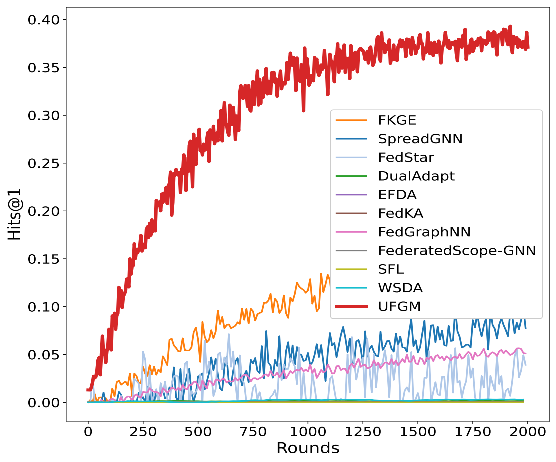
Evaluation metrics. By following the same settings in two representative graph matching models [91, 27], We employ a popular measure, $Hits@K$, to evaluate and compare our UFGM model to previous lines of work, where $Hits@K$ measures the proportion of correctly matched nodes ranked in the top-K list. A larger $Hits@K$ value indicates a better graph matching result. We use final $Hits@K$ to evaluate the quality of the federated federated learning algorithms. In addition, we plot the measure curves regarding $Hits@K$ and Loss Function Values (*Loss*) with increasing rounds to verify the convergence of different

Table 4.3: Final Performance on PPI

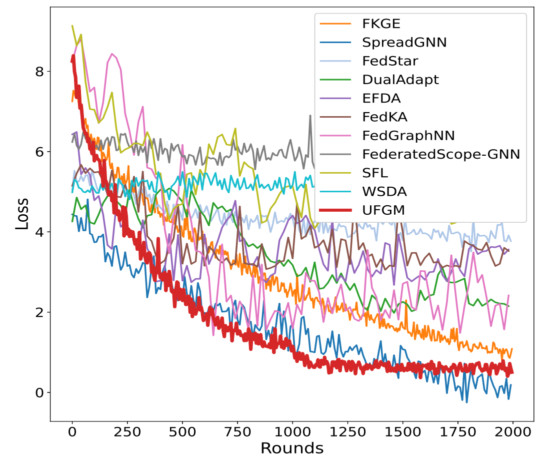
Type	Algorithm	*Hits*@1	*Hits*@5	*Hits*@10	*Hits*@50	*Loss*
Centralized Graph Matching	NextAlign	**0.430**	**0.512**	**0.571**	**0.635**	**2.149**
	NetTrans	0.379	0.439	0.447	0.496	1.611
	CPUGA	0.230	0.238	0.252	0.297	2.551
	ASAR-GM	0.199	0.229	0.252	0.337	1.410
	SIGMA	0.220	0.232	0.253	0.262	1.330
	SeedGNN	0.319	0.340	0.342	0.388	2.919
Federated Domain Adaption	DualAdapt	0.001	0.002	0.002	0.002	2.049
	EFDA	0.001	0.001	0.002	0.002	3.427
	WSDA	0.003	0.005	0.007	0.011	5.129
	FedKA	0.001	0.001	0.010	0.013	3.715
Federated Graph Learning	FedGraphNN	0.051	0.100	0.116	0.161	3.120
	FKGE	0.177	0.205	0.222	0.250	1.086
	SpreadGNN	0.078	0.146	0.175	0.192	**0.189**
	SFL	0.000	0.000	0.000	0.001	4.332
	FederatedScope-GNN	0.000	0.000	0.000	0.001	5.611
	FedStar	0.039	0.071	0.105	0.136	3.770
	UFGM	0.371	0.440	0.411	0.459	0.501

federated learning methods: [244, 259, 253, 271, 245, 285]. A smaller Loss score shows a better federated learning result.

Final *Hits*@*K* and *Loss* on SNS and PPI. Tables 4.2 and 4.3 show the quality of six centralized graph matching, six federated graph learning, and four federated domain adaption algorithms over SNS and PPI respectively. We have observed that our UFGM federated graph matching solution outperforms all the competitors of federated graph learning and federated domain adaption in most experiments. UFGM achieves the highest $Hits@K$ values (> 0.771 over SNS and > 0.371 on PPI respectively) and the lowest $Loss$ values ($= 0.659$ over SNS and $= 0.501$ on PPI respectively), which are better than other ten baseline methods in all tests. In addition, the $Hits@K$ scores achieved by UFGM is close or much better than the centralized graph matching method. Compared with the best centralized graph matching method, NextAlign, the $Hits@1$, $Hits@5$, $Hits@10$, and $Hits@50$ scores by UFGM are only 15.3% lower respectively. A reasonable explanation is that the combination of graphlet feature extraction, separate trust region, and pseudo supervised learning is able to achieve higher matching accuracy and faster convergence in federated settings. In addition, the promising performance of UFGM over both datasets implies that UFGM has great potential as a general federated graph matching solution over federated datasets.

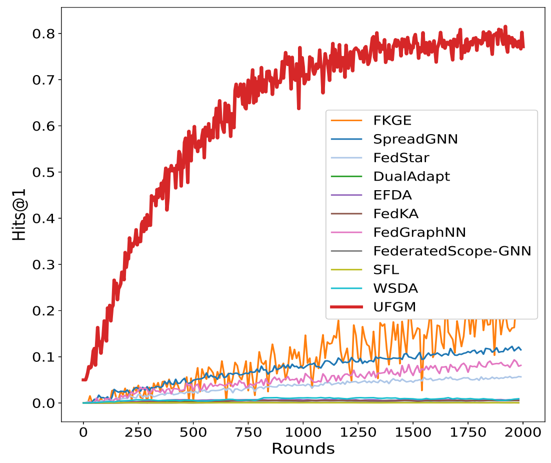


(a) *Hits*@1

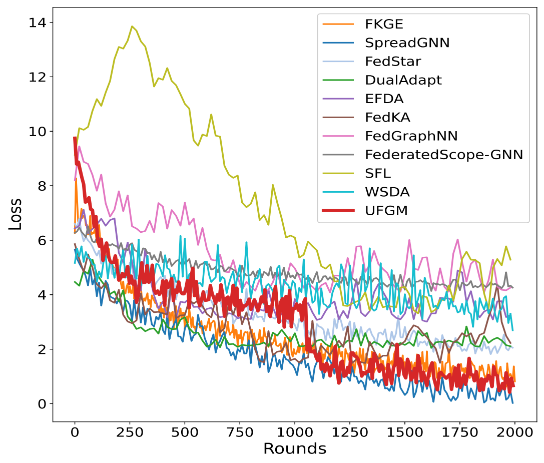


(b) *Loss*

Figure 4.1: Convergence on SNS



(a) *Hits*@1

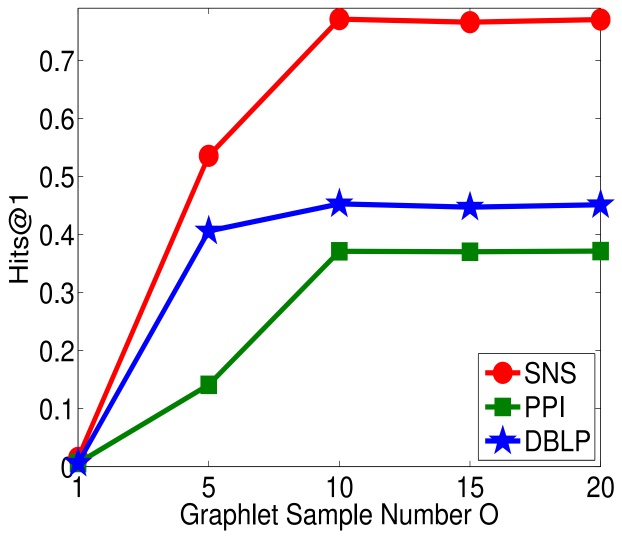


(b) *Loss*

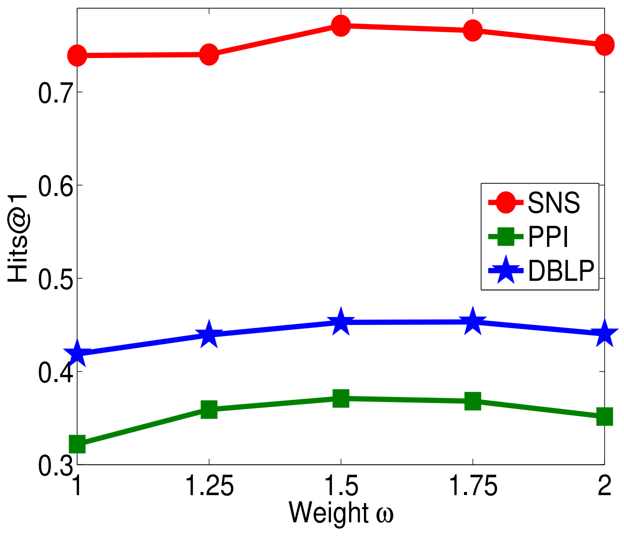
Figure 4.2: Convergence on PPI

Hits@K **Convergence on SNS and PPI.** Figures 4.1 and 4.2 exhibit the *Hits@K* curves of eleven federated learning models for graph matching over SNS and PPI respectively. It is obvious that the performance curves by federated learning algorithms initially keep increasing with training rounds and remains relatively stable when the curves are beyond convergence points, i.e., turning points from a sharp *Hits@K* increase to a flat curve. This phenomenon indicates that most federated learning algorithms are able to converge to the invariant solutions after enough training rounds. However, among six federated graph learning and four federated domain adaption approaches, our UFGM method can significantly speedup the convergence on two datasets in most experiments, showing the superior performance of UFGM in federated settings.

Loss **Convergence on SNS and PPI.** Figures 4.1 and 4.2 also present the *Loss* curves achieved by eleven federated learning models on two datasets respectively. We have observed obvious that the reverse trends, in comparison with the *Hits@K* curves. In most experiments, our UFGM is able to achieve the fastest convergence, especially, UFGM can converge around 1,000 training rounds and then always keep stable on two datasets. A reasonable explanation is that UFGM fully utilizes the proposed graphlet feature extraction techniques to generate the pseudo training data and employ the strength of supervised graph matching for accelerating the training convergence.



(a) O



(b) ω

Figure 4.3: Final $Hits@1$ with Varying Parameters

Impact of graphlet sample numbers. Figure 4.3 (a) measures the performance effect of sampled graphlet numbers in the Monte Carlo Markov Chain sampling for graphlet enumeration and estimation by varying O from 10 to 1,000. We have witnessed the performance curves by UFGM initially increase quickly and then become stable when O continuously increases. Initially, a large O can help utilize the strength of effective graphlet feature extraction for generating the pseudo training data for tackling the dilemma of unsupervised graph matching in federated setting and employing the strength of supervised graph matching. Later on, when O continues to increase and goes beyond some thresholds, the performance curves become stable. A rational guess is that after the enough graphlet features have been already extracted at a certain threshold and considered in the FGM training, our UFGM model is able to generate a good graph matching result. When O continuously increases, this does not affect the performance of graph matching any more.

Impact of weight ω between two types of weak quasi-Newton conditions. Figures 4.3 (b) shows the influence of weight of two types of weak quasi-Newton conditions in our UFGM model by varying it from 1 to 2. It is observed that the performance initially raises when the ω increases. Intuitively, a large ω can help the algorithm well balance two types of weak quasi-Newton conditions and thus help improve the quality of separate trust region and graph matching. Later on, the performance curves decrease quickly when the ω continuously increases. A reasonable explanation is that a too large ω may ruin the first type of weak quasi-Newton condition and miss the optimal solution in the search process. Thus, it is important to determine the optimal ω for separate trust region.

4.6 Related Work

Centralized Graph Matching. Graph matching, also well known as network alignment, which aims to identify the same entities (i.e., nodes) across multiple graphs, has been a heated topic in recent years [15, 87, 81, 99, 169, 35, 49, 27, 66, 144, 68]. Research activities can be classified into three broad categories. (1) Topological structure-based techniques, which rely on only the structural information of nodes to match two or multiple graphs, including DPMC [81], ZAC [284], GRAMPA [228], CONE-Align [11], DeepMatching [282], Exact Graph Matching [269], qc-DGM [230], OTTER [288], IA-GM [305], D-GAP [255], CPUGA [263], CAPER [310]; (2) Structure and/or attribute-based approaches, which utilize highly discriminative structure and attribute features for ensuring the matching effectiveness, such as gsaNA [91], REGAL [35], SNNA [49], CENALP [21], GAlign [38], Deep Graph Matching Consensus [27], CIE [94], RE [309], Meta-NA [307], EAGM [267], DLGM [297], SIGMA [252], SCGM [251], and Grad-Align+ [262]; (3) Heterogeneous methods employ heterogeneous structural, content, spatial, and temporal features to further improve the matching performance, including SCAN-PS [203], MNA [155], HYDRA [163], COSNET [207], Factoid Embedding [197], DPLink [144], DETA [258]. BANANA [68], SAUIL [266]. GCAN [241], and Deep Multi-Graph Matching [295]; Several papers review key achievements of graph matching across online information networks including state-of-the-art algorithms, evaluation metrics, representative datasets, and empirical analysis [179, 189, 89, 301, 232]. It has been widely applied to many real-world applications, including protein network alignment in bioinformatics [165, 189], user account linking in multiple social networks[179, 169, 144], object matching in computer vision [27, 284, 286, 294], knowledge translation in multilingual knowledge bases [311, 222, 254, 231, 311, 293] and text matching [226].

Federated Graph Learning. With the increasing privacy awareness, commercial competition, and regulation restrictions, real-world graph data is often generated locally and remains distributed graphs of multiple data silos among a large number of clients [306, 223, 298]. Federated graph learning (FGL) is a promising paradigm that enables collaborative training of shared machine learning models over large-scale distributed graph data, while preserving privacy of local data. Based on how graph data can be distributed across clients, existing FGL techniques on machine unlearning can be broadly classified into three categories below. (1) Graph-level FGL: each client possesses a set of graphs and all clients collaborate to train a shared model to predict graph properties, including [291, 233, 300, 227, 278, 237, 268]. Typical graph-level FGL task is graph classification/regression, which have been applied multiple domains, such as molecular property prediction [291, 236] and brain network analysis [219]; (2) Subgraph-level FL: each client contains a subgraph of a global graph, a part of node features, and a part of FGL model [299, 260, 281, 224, 218, 292, 304, 250, 312, 279]. The clients aim to collaboratively train a global model with the partial features and subgraphs to predict node properties. Typical graph-level FGL task is node classification and link prediction; (3) Node-level FGL: the clients are connected by a graph and thus each of them is treated as a node [248, 257, 220, 273]. Namely, the clients, rather than the data, are graph-structured. For example, each client performs learning with its own data and they exchange data through the communication graph [248, 257]. The server maintains the graph structure and uses a GNN to aggregate information (either models or data) collected from the clients [220, 273].

A recent work studied the problem of federated knowledge graphs embedding with a byproduct of knowledge graph alignment [264]. It exploits adversarial generation between pairs of knowledge graphs to translate identical entities and relations of different domains into near embedding spaces. To our best knowledge, this workThis work is the first to has the potential to tackle the problem of general federated graph matching. However, it is a supervised learning method with aligned entities and relations as training data. In addition, it is possible that neural models may memorize inputs and reconstruct inputs from corresponding outputs [221]. The method exchanges the embeddings of entities and relations between clients and server. Adversarial samples and gradients are interchanged among the clients. Although a host client cannot access the embeddings of the other's, the exchange of translational mapping matrices (1.e., the gradients in the generators of the other clients) makes it possible for the host client to reconstruct the former's embeddings with the inverse of translational mapping matrices. This work is the first to offer an unsupervised federated graph matching solution for inferring matched node pairs on different graphs across clients while maintaining the privacy requirement of federated learning, by leveraging the graphlet theory and trust region optimization.

4.7 Conclusions

In this work, we have proposed an unsupervised federated graph matching algorithm. First, an approximate graphlet enumeration method is proposed to capture nodes' graphlet features to generate pseudo matched node pairs as pseudo training data. Second, a separate trust region algorithm is proposed for pseudo supervised federated graph matching while maintaining the privacy constraints. Finally, empirical evaluation on real datasets demonstrates the superior performance of our UFGM.

Chapter 5

Conclusions

In this dissertation, we studied the impact of adversarial perturbation to the task of graph matching under various scenarios as while as the possibility of mitigating such influence via designed defensive mechanism. In our first work, we developed an attack that tries to confuse the trained neural model by pushing the target nodes into areas of relatively higher densities. Combined with meta-learnt starting point, the attack is demonstrated empirically to be effective against graph matching. We also found that neural models adversarial trained using examples generated by the above mentioned attack scheme was able to gain extra robustness against future attacks. In our second work, we investigated the possibility of providing a provable and certifiable robustness guarantee while maintaining the expressiveness of the neural model. A range of preferred value of norm constraints of the parameters are provided via theoretical analysis. We observed the coexistence of reasonable accuracy and robustness guarantee in the constrained model equipped designed activation functions combined and derived norm constraints. In our last work, we studied graph matching in the context of Federated Learning. We deconstructed the conflict of between the graph matching algorithm needing simultaneous knowledge of two or more graphs and the privacy constrains preventing local user from sharing their own graphs. An secure algorithm is developed for conducting graph matching in a Federated Learning scheme. The potential threats during the process, namely probing attacks from the locals and the reconstruction attacks from the server, are evaluated and dealt with anti-order preserving encryption and transformation validations respectively. At the end, we would like to conclude that graph matching does exhibit certain vulnerabilities in various adversarial scenarios, but these vulnerabilities are by no means insoluble.

Bibliography

[1] https://snap.stanford.edu/data/Oregon-2.html.

[2] https://snap.stanford.edu/data/as-Caida.html.

[3] http://dblp.uni-trier.de/xml/.

[4] https://chrsmrrs.github.io/datasets/docs/datasets/.

[5] Al-Rfou, R., Perozzi, B., and Zelle, D. DDGK: learning graph representations for deep divergence graph kernels. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pp. 37–48, 2019.

[6] Anil, C., Lucas, J., and Grosse, R. B. Sorting out lipschitz function approximation. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pp. 291–301, 2019.

[7] Bao, X., Liu, L., Xiao, N., Zhou, Y., and Zhang, Q. Policy-driven autonomic configuration management for nosql. In *Proceedings of the 2015 IEEE International Conference on Cloud Computing (CLOUD'15)*, pp. 245–252, New York, NY, June 27-July 2 2015.

[8] Bjorck, A. and Bowie, C. An iterative algorithm for computing the best estimate of an orthogonal matrix. *SIAM J. Numer. Anal.*, 8:358–364, 1971.

[9] Bojchevski, A. and Günnemann, S. Adversarial attacks on node embeddings via graph poisoning. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pp. 695–704, 2019.

[10] Byers, R. and Xu, H. A new scaling for newton's iteration for the polar decomposition and its backward stability. *SIAM J. Matrix Anal. Appl.*, 30(2):822–843, 2008.

[11] Chen, X., Heimann, M., Vahedian, F., and Koutra, D. Cone-align: Consistent network alignment with proximity-preserving node embedding. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pp. 1985–1988, 2020.

[12] Cheng, H., Lo, D., Zhou, Y., Wang, X., and Yan, X. Identifying bug signatures using discriminative graph mining. In *Proceedings of the 18th International Symposium on Software Testing and Analysis (ISSTA'09)*, pp. 141–152, Chicago, IL, July 19-23 2009.

[13] Cheng, H., Zhou, Y., and Yu, J. X. Clustering large attributed graphs: A balance between structural and attribute similarities. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 5(2):1–33, 2011.

[14] Cheng, H., Zhou, Y., Huang, X., and Yu, J. X. Clustering large attributed information networks: An efficient incremental computing approach. *Data Mining and Knowledge Discovery (DMKD)*, 25(3):450–477, 2012.

[15] Chu, X., Fan, X., Yao, D., Zhu, Z., Huang, J., and Bi, J. Cross-network embedding for multi-network alignment. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pp. 273–284, 2019.

[16] Cissé, M., Bojanowski, P., Grave, E., Dauphin, Y. N., and Usunier, N. Parseval networks: Improving robustness to adversarial examples. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pp. 854–863, 2017.

[17] Cohen, J. E. J., Huster, T., and Cohen, R. Universal lipschitz approximation in bounded depth neural networks. *CoRR*, abs/1904.04861, 2019.

[18] Dai, H., Li, H., Tian, T., Huang, X., Wang, L., Zhu, J., and Song, L. Adversarial attack on graph structured data. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pp. 1123–1132, 2018.

[19] Dai, Q., Shen, X., Zhang, L., Li, Q., and Wang, D. Adversarial training methods for network embedding. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pp. 329–339, 2019.

[20] Deng, L. The MNIST database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Process. Mag.*, 29(6):141–142, 2012.

[21] Du, X., Yan, J., and Zha, H. Joint link prediction and network alignment via cross-graph embedding. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pp. 2251–2257, 2019.

[22] Elinas, P., Bonilla, E. V., and Tiao, L. Variational inference for graph convolutional networks in the absence of graph data and adversarial settings. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, Online*, December 6-12 2020.

[23] Entezari, N., Al-Sayouri, S., Darvishzadeh, A., and Papalexakis, E. All you need is low (rank): Defending against adversarial attacks on graphs. In *Proceedings of the 13th ACM International Conference on Web Search and Data Mining, WSDM 2020, Houston, TX, February 3-7, 2020*, 2020.

[24] Fan, S., Wang, X., Shi, C., Lu, E., Lin, K., and Wang, B. One2multi graph autoencoder for multi-view graph clustering. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pp. 3070–3076, 2020.

[25] Fazlyab, M., Robey, A., Hassani, H., Morari, M., and Pappas, G. J. Efficient and accurate estimation of lipschitz constants for deep neural networks. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pp. 11423–11434, 2019.

[26] Feng, W., Zhang, J., Dong, Y., Han, Y., Luan, H., Xu, Q., Yang, Q., Kharlamov, E., and Tang, J. Graph random neural networks for semi-supervised learning on graphs. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, Online*, December 6-12 2020.

[27] Fey, M., Lenssen, J. E., Morris, C., Masci, J., and Kriege, N. M. Deep graph matching consensus. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*, 2020.

[28] Fu, D., Xu, Z., Li, B., Tong, H., and He, J. A view-adversarial framework for multi-view network embedding. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pp. 2025–2028, 2020.

[29] Fu, Y., Wang, P., Du, J., Wu, L., and Li, X. Efficient region embedding with multi-view spatial networks: A perspective of locality-constrained spatial autocorrelations. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pp. 906–913, 2019.

[30] Gander, W. Algorithms for the polar decomposition. *SIAM J. Scientific Computing*, 11(6):1102–1115, 1990.

[31] Gao, Z., Hu, R., and Gong, Y. Certified robustness of graph classification against topology attack with randomized smoothing. In *2020 IEEE Global Communications Conference, GLOBECOM 2020, Taipei, Taiwan, December 8-10, 2020*, 2020.

[32] Gil, A., Segura, J., and Temme, N. M. *Numerical methods for special functions.* SIAM, 2007.

[33] Goswami, S., Pokhrel, A., Lee, K., Liu, L., Zhang, Q., and Zhou, Y. Graphmap: Scalable iterative graph processing using nosql. *The Journal of Supercomputing (TJSC)*, 76(9):6619–6647, 2020.

[34] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 5767–5777, 2017.

[35] Heimann, M., Shen, H., Safavi, T., and Koutra, D. REGAL: representation learning-based graph alignment. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*, pp. 117–126, 2018.

[36] Higham, N. J. and Schreiber, R. S. Fast polar decomposition of an arbitrary matrix. *SIAM J. Sci. Comput.*, 11(4):648–655, 1990.

[37] Higham, N. J., Mackey, D. S., Mackey, N., and Tisseur, F. Computing the polar decomposition and the matrix sign decomposition in matrix groups. *SIAM J. Matrix Anal. Appl.*, 25(4):1178–1192, 2004.

[38] Huynh, T. T., Tong, V. V., Nguyen, T. T., Yin, H., Weidlich, M., and Hung, N. Q. V. Adaptive network alignment with unsupervised and multi-order convolutional networks. In *36th IEEE International Conference on Data Engineering, ICDE 2020, Dallas, TX, USA, April 20-24, 2020*, pp. 85–96, 2020.

[39] Jia, J., Wang, B., Cao, X., and Gong, N. Z. Certified robustness of community detection against adversarial structural perturbation via randomized smoothing. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pp. 2718–2724, 2020.

[40] Jin, H., Shi, Z., Peruri, A., and Zhang, X. Certified robustness of graph convolution networks for graph classification under topological attacks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020 (NeurIPS'20)*, Online, December 6-12 2020a.

[41] Jin, R., Li, D., Gao, J., Liu, Z., Chen, L., and Zhou, Y. Towards a better understanding of linear models for recommendation. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'21)*, Virtual Event, August 14-18 2021.

[42] Jin, W., Ma, Y., Liu, X., Tang, X., Wang, S., and Tang, J. Graph structure learning for robust graph neural networks. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pp. 66–74, 2020b.

[43] Kriege, N. M., Neumann, M., Morris, C., Kersting, K., and Mutzel, P. A unifying view of explicit and implicit feature maps of graph kernels. *Data Min. Knowl. Discov.*, 33 (6):1505–1547, 2019.

[44] Krishnan, V., Makdah, A. A. A., and Pasqualetti, F. Lipschitz bounds and provably robust training by laplacian smoothing. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[45] Krizhevsky, A. Learning multiple layers of features from tiny images. *Technical Report*, 2009.

[46] Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*, pp. 1106–1114, 2012.

[47] Lee, K., Liu, L., Tang, Y., Zhang, Q., and Zhou, Y. Efficient and customizable data partitioning framework for distributed big rdf data processing in the cloud. In *Proceedings of the 2013 IEEE International Conference on Cloud Computing (CLOUD'13)*, pp. 327–334, Santa Clara, CA, June 27-July 2 2013.

[48] Lee, K., Liu, L., Schwan, K., Pu, C., Zhang, Q., Zhou, Y., Yigitoglu, E., and Yuan, P. Scaling iterative graph computations with graphmap. In *Proceedings of the 27th IEEE international conference for High Performance Computing, Networking, Storage and Analysis (SC'15)*, pp. 57:1–57:12, Austin, TX, November 15-20 2015.

[49] Li, C., Wang, S., Wang, Y., Yu, P. S., Liang, Y., Liu, Y., and Li, Z. Adversarial learning for weakly-supervised social network alignment. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pp. 996–1003, 2019a.

[50] Li, Q., Haque, S., Anil, C., Lucas, J., Grosse, R. B., and Jacobsen, J. Preventing gradient attenuation in lipschitz constrained convolutional networks. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pp. 15364–15376, 2019b.

[51] Liu, Y., He, L., Cao, B., Yu, P. S., Ragin, A. B., and Leow, A. D. Multi-view multi-graph embedding for brain network clustering analysis. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pp. 117–124, 2018.

[52] Logins, A., Li, Y., and Karras, P. On the robustness of cascade diffusion under node attacks. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pp. 2711–2717, 2020.

[53] Luo, D., Bian, Y., Yan, Y., Liu, X., Huan, J., and Zhang, X. Local community detection in multiple networks. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pp. 266–274, 2020.

[54] Ma, G., He, L., Lu, C.-T., Shao, W., Yu, P. S., Leow, A. D., and Ragin, A. B. Multi-view clustering with graph embedding for connectome analysis. In *Proc. 2017 Int.*

Conf. Information and Knowledge Management (CIKM'17), pp. 127–136, Singapore, November 6-10 2017.

[55] Ma, N., Bu, J., Yang, J., Zhang, Z., Yao, C., Yu, Z., Zhou, S., and Yan, X. Adaptive-step graph meta-learner for few-shot graph classification. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pp. 1055–1064, 2020a.

[56] Ma, T., Wang, H., Zhang, L., Tian, Y., and Al-Nabhan, N. Graph classification based on structural features of significant nodes and spatial convolutional neural networks. *Neurocomputing*, 423:639–650, 2021.

[57] Ma, Z., Xuan, J., Wang, Y. G., Li, M., and Liò, P. Path integral based convolution and pooling for graph neural networks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020b.

[58] Magelinski, T., Beskow, D. M., and Carley, K. M. Graph-hist: Graph classification from latent feature histograms with application to bot detection. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pp. 5134–5141, 2020.

[59] Man, T., Shen, H., Liu, S., Jin, X., and Cheng, X. Predict anchor links across social networks via an embedding approach. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pp. 1823–1829, 2016.

[60] Miller, B. A., Camurcu, M., Gomez, A. J., Chan, K., and Eliassi-Rad, T. Improving robustness to attacks against vertex classification. In *Proceedings of the 15th International Workshop on Mining and Learning with Graphs co-located with 24th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, MLG@KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, 2019.

[61] Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. Spectral normalization for generative adversarial networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.

[62] Oettershagen, L., Kriege, N. M., Morris, C., and Mutzel, P. Temporal graph kernels for classifying dissemination processes. In *Proceedings of the 2020 SIAM International Conference on Data Mining, SDM 2020, Cincinnati, Ohio, USA, May 7-9, 2020*, pp. 496–504, 2020.

[63] Palanisamy, B., Liu, L., Lee, K., Meng, S., Tang, Y., and Zhou, Y. Anonymizing continuous queries with delay-tolerant mix-zones over road networks. *Distributed and Parallel Databases (DAPD)*, 32(1):91–118, 2014.

[64] Palanisamy, B., Liu, L., Zhou, Y., and Wang, Q. Privacy-preserving publishing of multilevel utility-controlled graph datasets. *ACM Transactions on Internet Technology (TOIT)*, 18(2):24:1–24:21, 2018.

[65] Peng, H., Li, J., Gong, Q., Ning, Y., Wang, S., and He, L. Motif-matching based subgraph-level attentional convolutional network for graph classification. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pp. 5387–5394, 2020.

[66] Qin, K. K., Salim, F. D., Ren, Y., Shao, W., Heimann, M., and Koutra, D. G-CREWE: graph compression with embedding for network alignment. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pp. 1255–1264, 2020.

[67] Qu, M., Tang, J., Shang, J., Ren, X., Zhang, M., and Han, J. An attention-based collaboration framework for multi-view network representation learning. In *Proc. 2017 Int. Conf. Information and Knowledge Management (CIKM'17)*, pp. 1767–1776, Singapore, November 6-10 2017.

[68] Ren, F., Zhang, Z., Zhang, J., Su, S., Sun, L., Zhu, G., and Guo, C. BANANA: when behavior analysis meets social network alignment. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pp. 1438–1444, 2020.

[69] Ren, J., Zhou, Y., Jin, R., Zhang, Z., Dou, D., and Wang, P. Dual adversarial learning based network alignment. In *Proceedings of the 19th IEEE International Conference on Data Mining (ICDM'19)*, pp. 1288–1293, Beijing, China, November 8-11 2019.

[70] Rieck, B., Bock, C., and Borgwardt, K. M. A persistent weisfeiler-lehman procedure for graph classification. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pp. 5448–5458, 2019.

[71] Sharma, J. and Vasishtha, A. *Kirshna's Real Analysis: (General), Thirty Eighth Edition*. Krishna Prakashan Media, 2010.

[72] Sokolic, J., Giryes, R., Sapiro, G., and Rodrigues, M. R. D. Robust large margin deep neural networks. *IEEE Trans. Signal Process.*, 65(16):4265–4280, 2017.

[73] Su, Z., Liu, L., Li, M., Fan, X., and Zhou, Y. Servicetrust: Trust management in service provision networks. In *Proceedings of the 10th IEEE International Conference on Services Computing (SCC'13)*, pp. 272–279, Santa Clara, CA, June 27-July 2 2013.

[74] Su, Z., Liu, L., Li, M., Fan, X., and Zhou, Y. Reliable and resilient trust management in distributed service provision networks. *ACM Transactions on the Web (TWEB)*, 9 (3):1–37, 2015.

[75] Sun, Y., Wang, S., Hsieh, T., Tang, X., and Honavar, V. G. MEGAN: A generative adversarial network for multi-view network embedding. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pp. 3527–3533, 2019.

[76] Tang, H., Ma, G., Chen, Y., Guo, L., Wang, W., Zeng, B., and Zhan, L. Adversarial attack on hierarchical graph pooling neural networks. *CoRR*, abs/2005.11560, 2020a.

[77] Tang, X., Li, Y., Sun, Y., Yao, H., Mitra, P., and Wang, S. Transferring robustness for graph neural network against poisoning attacks. In *Proceedings of the 13th ACM International Conference on Web Search and Data Mining, WSDM 2020, Houston, TX, February 3-7, 2020*, 2020b.

[78] Togninalli, M., Ghisu, M. E., Llinares-López, F., Rieck, B., and Borgwardt, K. M. Wasserstein weisfeiler-lehman graph kernels. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 6436–6446, 2019.

[79] Tsuzuku, Y., Sato, I., and Sugiyama, M. Lipschitz-margin training: Scalable certification of perturbation invariance for deep neural networks. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*, pp. 6542–6551, 2018.

[80] Wang, R., Yan, J., and Yang, X. Graduated assignment for joint multi-graph matching and clustering with application to unsupervised graph matching network learning. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020a.

[81] Wang, T., Jiang, Z., and Yan, J. Multiple graph matching and clustering via decayed pairwise matching composition. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pp. 1660–1667, 2020b.

[82] Weibull, W. A statistical distribution function of wide applicability. *Journal of Applied Mechanics*, 18:293–297, 1951.

[83] Wu, J., He, J., and Xu, J. Demo-net: Degree-specific graph neural networks for node and graph classification. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pp. 406–415, 2019.

[84] Wu, S., Li, Y., Zhang, D., Zhou, Y., and Wu, Z. Diverse and informative dialogue generation with context-specific commonsense knowledge awareness. In *Proceedings of*

the 58th Annual Meeting of the Association for Computational Linguistics, (ACL'20), pp. 5811–5820, Online, July 5-10 2020.

[85] Wu, S., Li, Y., Zhang, D., Zhou, Y., and Wu, Z. Topicka: Generating commonsense knowledge-aware dialogue responses towards the recommended topic fact. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, (IJCAI'20)*, pp. 3766–3772, Online, January 7-15 2021a.

[86] Wu, S., Wang, M., Zhang, D., Zhou, Y., Li, Y., and Wu, Z. Knowledge-aware dialogue generation via hierarchical infobox accessing and infobox-dialogue interaction graph network. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence, (IJCAI'21)*, Online, August 21-26 2021b.

[87] Xu, H., Luo, D., Zha, H., and Carin, L. Gromov-wasserstein learning for graph matching and node embedding. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pp. 6932–6941, 2019a.

[88] Xu, K., Chen, H., Liu, S., Chen, P., Weng, T., Hong, M., and Lin, X. Topology attack and defense for graph neural networks: An optimization perspective. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pp. 3961–3967, 2019b.

[89] Yan, J., Yang, S., and Hancock, E. R. Learning for graph matching and related combinatorial optimization problems. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pp. 4988–4996, 2020.

[90] Yanardag, P. and Vishwanathan, S. V. N. Deep graph kernels. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*, pp. 1365–1374, 2015.

[91] Yasar, A. and Çatalyürek, Ü. V. An iterative global structure-assisted labeled network aligner. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pp. 2614–2623, 2018.

[92] Yoshida, Y. and Miyato, T. Spectral norm regularization for improving the generalizability of deep learning. *CoRR*, abs/1705.10941, 2017.

[93] You, Y., Chen, T., Sui, Y., Chen, T., Wang, Z., and Shen, Y. Graph contrastive learning with augmentations. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020 (NeurIPS'20)*, Online, December 6-12 2020.

[94] Yu, T., Wang, R., Yan, J., and Li, B. Learning deep graph matching with channel-independent embedding and hungarian attention. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*, 2020a.

[95] Yu, T., Yan, J., and Li, B. Determinant regularization for gradient-efficient graph matching. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pp. 7121–7130, 2020b.

[96] Yu, Y., Xu, G., Jiang, M., Zhu, H., Dai, D., and Yan, H. Joint transformation learning via the $l_{2,1}$-norm metric for robust graph matching. *IEEE Trans. Cybern.*, 51(2):521–533, 2021.

[97] Zhang, G., Zhou, Y., Wu, S., Zhang, Z., and Dou, D. Cross-lingual entity alignment with adversarial kernel embedding and adversarial knowledge translation. *CoRR*, abs/2104.07837, 2021.

[98] Zhang, L. and Lu, H. A feature-importance-aware and robust aggregator for GCN. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pp. 1813–1822, 2020.

[99] Zhang, S. and Tong, H. FINAL: fast attributed network alignment. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2016, San Francisco, CA, USA, August 13-17, 2016*, pp. 1345–1354, 2016.

[100] Zhang, S., Tong, H., Maciejewski, R., and Eliassi-Rad, T. Multilevel network alignment. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pp. 2344–2354, 2019.

[101] Zhang, X. and Zitnik, M. Gnnguard: Defending graph neural networks against adversarial attacks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, Online*, December 6-12 2020.

[102] Zhang, Z., Zhang, Z., Zhou, Y., Shen, Y., Jin, R., and Dou, D. Adversarial attacks on deep graph matching. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020 (NeurIPS'20)*, Virtual, December 6-12 2020.

[103] Zhao, Q. and Wang, Y. Learning metrics for persistence-based summaries and applications for graph classification. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 9855–9866, 2019.

[104] Zheng, C., Zong, B., Cheng, W., Song, D., Ni, J., Yu, W., Chen, H., and Wang, W. Robust graph representation learning via neural sparsification. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2019, Online*, 2020.

[105] Zhou, F., Liu, L., Zhang, K., Trajcevski, G., Wu, J., and Zhong, T. Deeplink: A deep learning approach for user identity linkage. In *2018 IEEE Conference on Computer Communications, INFOCOM 2018, Honolulu, HI, USA, April 16-19, 2018*, pp. 1313–1321, 2018a.

[106] Zhou, K. and Vorobeychik, Y. Robust collective classification against structural attacks. In *Proceedings of the Thirty-Sixth Conference on Uncertainty in Artificial Intelligence, UAI 2020, virtual online, August 3-6, 2020*, pp. 119, 2020.

[107] Zhou, K., Michalak, T. P., and Vorobeychik, Y. Adversarial robustness of similarity-based link prediction. In *IEEE International Conference on Data Mining, ICDM 2019, Beijing, China, November 8-11, 2019*, 2019a.

[108] Zhou, Y. *Innovative Mining, Processing, and Application of Big Graphs.* PhD thesis, Georgia Institute of Technology, Atlanta, GA, USA, 2017.

[109] Zhou, Y. and Liu, L. Clustering analysis in large graphs with rich attributes. In Holmes, D. E. and Jain, L. C. (eds.), *Data Mining: Foundations and Intelligent Paradigms: Volume 1: Clustering, Association and Classification.* Springer, 2011.

[110] Zhou, Y. and Liu, L. Social influence based clustering of heterogeneous information networks. In *Proceedings of the 19th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'13)*, pp. 338–346, Chicago, IL, August 11-14 2013.

[111] Zhou, Y. and Liu, L. Activity-edge centric multi-label classification for mining heterogeneous information networks. In *Proceedings of the 20th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'14)*, pp. 1276–1285, New York, NY, August 24-27 2014.

[112] Zhou, Y. and Liu, L. Social influence based clustering and optimization over heterogeneous information networks. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 10(1):1–53, 2015.

[113] Zhou, Y. and Liu, L. Approximate deep network embedding for mining large-scale graphs. In *Proceedings of the 2019 IEEE International Conference on Cognitive Machine Intelligence (CogMI'19)*, pp. 53–60, Los Angeles, CA, December 12-14 2019.

[114] Zhou, Y., Cheng, H., and Yu, J. X. Graph clustering based on structural/attribute similarities. *Proceedings of the VLDB Endowment (PVLDB)*, 2(1):718–729, 2009.

[115] Zhou, Y., Cheng, H., and Yu, J. X. Clustering large attributed graphs: An efficient incremental approach. In *Proceedings of the 10th IEEE International Conference on Data Mining (ICDM'10)*, pp. 689–698, Sydney, Australia, December 14-17 2010.

[116] Zhou, Y., Liu, L., Perng, C.-S., Sailer, A., Silva-Lepe, I., and Su, Z. Ranking services by service network structure and service attributes. In *Proceedings of the 20th International Conference on Web Service (ICWS'13)*, pp. 26–33, Santa Clara, CA, June 27-July 2 2013.

[117] Zhou, Y., Seshadri, S., Chiu, L., and Liu, L. Graphlens: Mining enterprise storage workloads using graph analytics. In *Proceedings of the 2014 IEEE International Congress on Big Data (BigData'14)*, pp. 1–8, Anchorage, AK, June 27-July 2 2014.

[118] Zhou, Y., Liu, L., and Buttler, D. Integrating vertex-centric clustering with edge-centric clustering for meta path graph analysis. In *Proceedings of the 21st ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'15)*, pp. 1563–1572, Sydney, Australia, August 10-13 2015a.

[119] Zhou, Y., Liu, L., Lee, K., Pu, C., and Zhang, Q. Fast iterative graph computation with resource aware graph parallel abstractions. In *Proceedings of the 24th ACM Symposium on High-Performance Parallel and Distributed Computing (HPDC'15)*, pp. 179–190, Portland, OR, June 15-19 2015b.

[120] Zhou, Y., Liu, L., Lee, K., and Zhang, Q. Graphtwist: Fast iterative graph computation with two-tier optimizations. *Proceedings of the VLDB Endowment (PVLDB)*, 8(11): 1262–1273, 2015c.

[121] Zhou, Y., Liu, L., Pu, C., Bao, X., Lee, K., Palanisamy, B., Yigitoglu, E., and Zhang, Q. Clustering service networks with entity, attribute and link heterogeneity. In *Proceedings of the 22nd International Conference on Web Service (ICWS'15)*, pp. 257–264, New York, NY, June 27-July 2 2015d.

[122] Zhou, Y., Liu, L., Seshadri, S., and Chiu, L. Analyzing enterprise storage workloads with graph modeling and clustering. *IEEE Journal on Selected Areas in Communications (JSAC)*, 34(3):551–574, 2016.

[123] Zhou, Y., Amimeur, A., Jiang, C., Dou, D., Jin, R., and Wang, P. Density-aware local siamese autoencoder network embedding with autoencoder graph clustering. In *Proceedings of the 2018 IEEE International Conference on Big Data (BigData'18)*, pp. 1162–1167, Seattle, WA, December 10-13 2018b.

[124] Zhou, Y., Wu, S., Jiang, C., Zhang, Z., Dou, D., Jin, R., and Wang, P. Density-adaptive local edge representation learning with generative adversarial network multi-label edge classification. In *Proceedings of the 18th IEEE International Conference on Data Mining (ICDM'18)*, pp. 1464–1469, Singapore, November 17-20 2018c.

[125] Zhou, Y., Jiang, C., Zhang, Z., Dou, D., Jin, R., and Wang, P. Integrating local vertex/edge embedding via deep matrix fusion and siamese multi-label classification. In *Proceedings of the 2019 IEEE International Conference on Big Data (BigData'19)*, pp. 1018–1027, Los Angeles, CA, December 9-12 2019b.

[126] Zhou, Y., Ling Liu, Qi Zhang, K. L., and Palanisamy, B. Enhancing collaborative filtering with multi-label classification. In *Proceedings of the 2019 International Conference on Computational Data and Social Networks (CSoNet'19)*, pp. 323–338, Ho Chi Minh City, Vietnam, November 18-20 2019c.

[127] Zhou, Y., Ren, J., Wu, S., Dou, D., Jin, R., Zhang, Z., and Wang, P. Semi-supervised classification-based local vertex ranking via dual generative adversarial nets. In *Proceedings of the 2019 IEEE International Conference on Big Data (BigData'19)*, pp. 1267–1273, Los Angeles, CA, December 9-12 2019d.

[128] Zhou, Y., Liu, L., Lee, K., Palanisamy, B., and Zhang, Q. Improving collaborative filtering with social influence over heterogeneous information networks. *ACM Transactions on Internet Technology (TOIT)*, 20(4):36:1–36:29, 2020a.

[129] Zhou, Y., Ren, J., Dou, D., Jin, R., Zheng, J., and Lee, K. Robust meta network embedding against adversarial attacks. In *Proceedings of the 20th IEEE International Conference on Data Mining (ICDM'20)*, pp. 1448–1453, Sorrento, Italy, November 17-20 2020b.

[130] Zhou, Y., Ren, J., Jin, R., Zhang, Z., Dou, D., and Yan, D. Unsupervised multiple network alignment with multinominal gan and variational inference. In *Proceedings of the 2020 IEEE International Conference on Big Data (BigData'20)*, pp. 868–877, Atlanta, GA, December 10-13 2020c.

[131] Zhou, Y., Zhang, Z., Wu, S., Sheng, V., Han, X., Zhang, Z., and Jin, R. Robust network alignment via attack signal scaling and adversarial perturbation elimination. In *Proceedings of the 30th Web Conference (WWW'21)*, pp. 3884–3895, Virtual Event / Ljubljana, Slovenia, April 19-23 2021.

[132] Zhu, D., Zhang, Z., Cui, P., and Zhu, W. Robust graph convolutional networks against adversarial attacks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pp. 1399–1407, 2019.

[133] http://snap.stanford.edu/data/.

[134] L. Bertinetto, J. F. Henriques, P. H. S. Torr, and A. Vedaldi. Meta-learning with differentiable closed-form solvers. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019.

[135] H. Chang, Y. Rong, T. Xu, W. Huang, H. Zhang, P. Cui, W. Zhu, and J. Huang. A restricted black-box adversarial framework towards attacking graph embedding models. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, New Yrok, NY, USA, February 7 - 12, 2020*, 2020.

[136] J. Chen, Y. Wu, X. Xu, Y. Chen, H. Zheng, and Q. Xuan. Fast gradient attack on network embedding. *CoRR*, abs/1809.02797, 2018.

[137] J. Chen, L. Chen, Y. Chen, M. Zhao, S. Yu, Q. Xuan, and X. Yang. Ga-based q-attack on community detection. *IEEE Trans. Comput. Social Systems*, 6(3):491–503, 2019.

[138] J. Chen, Y. Chen, H. Zheng, S. Shen, S. Yu, D. Zhang, and Q. Xuan. MGA: momentum gradient attack on network. *CoRR*, abs/2002.11320, 2020.

[139] Y. Chen, Y. Nadji, A. Kountouras, F. Monrose, R. Perdisci, M. Antonakakis, and N. Vasiloglou. Practical attacks against graph-based clustering. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, pages 1125–1142, 2017a.

[140] Z. Chen, X. Yu, B. Song, J. Gao, X. Hu, and W. Yang. Community-based network alignment for large attributed network. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017, Singapore, November 06 - 10, 2017*, pages 587–596, 2017b.

[141] P. Dey and S. Medya. Manipulating node similarity measures in networks. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '20, Auckland, New Zealand, May 9-13, 2020*, 2019.

[142] B. Du and H. Tong. FASTEN: fast sylvester equation solver for graph mining. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pages 1339–1347, 2018.

[143] B. Du, S. Zhang, N. Cao, and H. Tong. FIRST: fast interactive attributed subgraph matching. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*, pages 1447–1456, 2017.

[144] J. Feng, M. Zhang, H. Wang, Z. Yang, C. Zhang, Y. Li, and D. Jin. Dplink: User identity linkage via deep neural network from heterogeneous mobility data. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 459–469, 2019.

[145] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 1126–1135, 2017.

[146] S. Fu, G. Wang, S. Xia, and L. Liu. Deep multi-granularity graph embedding for user identity linkage across social networks. *Knowl. Based Syst.*, 193:105301, 2020.

[147] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2010, Chia Laguna Resort, Sardinia, Italy, May 13-15, 2010*, pages 249–256, 2010.

[148] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[149] P. H. Guzzi and T. Milenkovic. Survey of local and global biological network alignment: the need to reconcile the two sides of the same coin. *Briefings in Bioinformatics*, 19 (3):472–481, 2018.

[150] N. L. Hjort and I. K. Glad. Nonparametric density estimation with a parametric start. *The Annals of Statistics*, 23(3):882–904, 1995.

[151] S. Hou, Y. Fan, Y. Zhang, Y. Ye, J. Lei, W. Wan, J. Wang, Q. Xiong, and F. Shao. α*Cyber*: Enhancing robustness of android malware detection system against adversarial attacks on heterogeneous graph based model. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, pages 609–618, 2019.

[152] T. T. Huynh, N. T. Toan, V. V. Tong, T. D. Hoang, D. C. Thang, N. Q. V. Hung, and A. Sattar. A comparative study on network alignment techniques. *Expert Syst. Appl.*, 140, 2020.

[153] H. Jiang. Uniform convergence rates for kernel density estimation. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 1694–1703, 2017.

[154] B. P. Kelley, R. Sharan, R. M. Karp, T. Sittler, D. E. Root, B. R. Stockwell, and T. Ideker. Conserved pathways within bacteria and yeast as revealed by global protein network alignment. *Proceedings of the National Academy of Sciences*, 100(20):11394–11399, 2003.

[155] X. Kong, J. Zhang, and P. S. Yu. Inferring anchor links across multiple heterogeneous social networks. In *22nd ACM International Conference on Information and Knowledge Management, CIKM'13, San Francisco, CA, USA, October 27 - November 1, 2013*, pages 179–188, 2013.

[156] R. Leibrandt and S. Günnemann. Making kernel density estimation robust towards missing values in highly incomplete multivariate data without imputation. In *Proceedings of the 2018 SIAM International Conference on Data Mining, SDM 2018, May 3-5, 2018, San Diego Marriott Mission Valley, San Diego, CA, USA*, pages 747–755, 2018.

[157] C. Li, S. Wang, P. S. Yu, L. Zheng, X. Zhang, Z. Li, and Y. Liang. Distribution distance minimization for unsupervised user identity linkage. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*, pages 447–456, 2018.

[158] C. Li, S. Wang, H. Wang, Y. Liang, P. S. Yu, Z. Li, and W. Wang. Partially shared adversarial learning for semi-supervised multi-platform user identity linkage. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, pages 249–258, 2019a.

[159] G. Li, L. Sun, Z. Zhang, P. Ji, S. Su, and P. S. Yu. Mc^2: Unsupervised multiple social network alignment. In *2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, December 9-12, 2019*, pages 1151–1156, 2019c.

[160] J. Li, H. Zhang, Z. Han, Y. Rong, H. Cheng, and J. Huang. Adversarial attack on community detection by hiding individuals. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 917–927, 2020.

[161] H. Liu, R. Socher, and C. Xiong. Taming MAML: efficient unbiased meta-reinforcement learning. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pages 4061–4071, 2019a.

[162] L. Liu, W. K. Cheung, X. Li, and L. Liao. Aligning users across social networks using network embedding. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 1774–1780, 2016.

[163] S. Liu, S. Wang, F. Zhu, J. Zhang, and R. Krishnan. HYDRA: large-scale social identity linkage via heterogeneous behavior modeling. In *International Conference on Management of Data, SIGMOD 2014, Snowbird, UT, USA, June 22-27, 2014*, pages 51–62, 2014.

[164] X. Liu, S. Si, X. Zhu, Y. Li, and C. Hsieh. A unified framework for data poisoning attack to graph-based semi-supervised learning. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2019, 8-14 December 2019, Vancouver, Canada*, pages 9777–9787, 2019b.

[165] Y. Liu, H. Ding, D. Chen, and J. Xu. Novel geometric approach for global alignment of PPI networks. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, AAAI 2017, February 4-9, 2017, San Francisco, California, USA.*, pages 31–37, 2017.

[166] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.

[167] E. Malmi, A. Gionis, and E. Terzi. Active network alignment: A matching-based approach. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017, Singapore, November 06 - 10, 2017*, pages 1687–1696, 2017.

[168] F. Masrour, P. Tan, and A. Esfahanian. OPTANE: an optimal transport algorithm for network alignment. In *ASONAM '19: International Conference on Advances in Social Networks Analysis and Mining, Vancouver, British Columbia, Canada, 27-30 August, 2019*, pages 448–451, 2019.

[169] X. Mu, F. Zhu, E. Lim, J. Xiao, J. Wang, and Z. Zhou. User identity linkage by latent user space modelling. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 1775–1784, 2016.

[170] X. Mu, W. Xie, R. K. Lee, F. Zhu, and E. Lim. Ad-link: An adaptive approach for user identity linkage. In *2019 IEEE International Conference on Big Knowledge, ICBK 2019, Beijing, China, November 10-11, 2019*, pages 183–190, 2019.

[171] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, pages 807–814, 2010.

[172] H. Nassar, N. Veldt, S. Mohammadi, A. Grama, and D. F. Gleich. Low rank spectral network alignment. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, pages 619–628, 2018.

[173] E. Parzen. On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076, 1962.

[174] A. Raghu, M. Raghu, S. Bengio, and O. Vinyals. Rapid learning or feature reuse? towards understanding the effectiveness of MAML. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*, 2020.

[175] V. Ravindra, H. Nassar, D. F. Gleich, and A. Grama. Rigid graph alignment. In *Complex Networks and Their Applications VIII - Volume 1 Proceedings of the Eighth International Conference on Complex Networks and Their Applications COMPLEX NETWORKS 2019, Lisbon, Portugal, December 10-12, 2019*, pages 621–632, 2019.

[176] Y. Ren, C. C. Aggarwal, and J. Zhang. Meta diagram based active social networks alignment. In *35th IEEE International Conference on Data Engineering, ICDE 2019, Macao, China, April 8-11, 2019*, pages 1690–1693, 2019b.

[177] S. R. Sain and D. W. Scott. On locally adaptive density estimation. *Journal of the American Statistical Association*, 91(436):1525–1534, 1996.

[178] Y. Shi, S. Wang, and Y. Han. Curls & whey: Boosting black-box adversarial attacks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 6519–6527, 2019.

[179] K. Shu, S. Wang, J. Tang, R. Zafarani, and H. Liu. User identity linkage across online social networks: A review. *SIGKDD Explorations*, 18(2):5–17, 2016.

[180] R. Singh, J. Xu, and B. Berger. Global alignment of multiple protein interaction networks with application to functional orthology detection. *Proceedings of the National Academy of Sciences*, 105(35):12763–12768, 2008. ISSN 0027-8424.

[181] S. Su, L. Sun, Z. Zhang, G. Li, and J. Qu. MASTER: across multiple social networks, integrate attribute and structure embedding for reconciliation. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden.*, pages 3863–3869, 2018.

[182] L. Sun, J. Wang, P. S. Yu, and B. Li. Adversarial attack and defense on graph data: A survey. *CoRR*, abs/1812.10528, 2018a.

[183] L. Sun, Z. Zhang, P. Ji, J. Wen, S. Su, and P. S. Yu. DNA: dynamic social network alignment. In *2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, December 9-12, 2019*, pages 1224–1231, 2019a.

[184] M. Sun, J. Tang, H. Li, B. Li, C. Xiao, Y. Chen, and D. Song. Data poisoning attack against unsupervised node embedding methods. *CoRR*, abs/1810.12881, 2018b.

[185] Y. Sun, S. Wang, X. Tang, T. Hsieh, and V. G. Honavar. Node injection attacks on graphs via reinforcement learning. *CoRR*, abs/1909.06543, 2019b.

[186] Y. Sun, S. Wang, X. Tang, T. Hsieh, and V. G. Honavar. Adversarial attacks on graph neural networks via node injections: A hierarchical reinforcement learning approach. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 673–683, 2020.

[187] T. Takahashi. Indirect adversarial attacks via poisoning neighbors for graph convolutional networks. In *2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, December 9-12, 2019*, pages 1395–1400, 2019.

[188] V. Vijayan and T. Milenkovic. Multiple network alignment via multimagna++. *IEEE/ACM Trans. Comput. Biology Bioinform.*, 15(5):1669–1682, 2018.

[189] V. Vijayan, S. Gu, E. T. Krebs, L. Meng, and T. Milenkovic. Pairwise versus multiple global network alignment. *IEEE Access*, 8:41961–41974, 2020.

[190] B. Wang and N. Z. Gong. Attacking graph-based classification via manipulating the graph structure. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019, London, UK, November 11-15, 2019*, pages 2023–2040, 2019.

[191] S. Wang, X. Li, Y. Ye, S. Feng, R. Y. K. Lau, X. Huang, and X. Du. Anchor link prediction across attributed networks via network embedding. *Entropy*, 21(3):254, 2019a.

[192] X. Wang, M. Cheng, J. Eaton, C. Hsieh, and S. F. Wu. Fake node attacks on graph convolutional networks. *CoRR*, abs/1810.10751, 2018.

[193] Y. Wang, C. Feng, L. Chen, H. Yin, C. Guo, and Y. Chu. User identity linkage across social networks via linked heterogeneous network embedding. *World Wide Web*, 22 (6):2611–2632, 2019b.

[194] M. Waniek, T. P. Michalak, M. J. Wooldridge, and T. Rahwan. Hiding individuals and communities in a social network. *Nature Human Behaviour*, 2:139–147, 2018.

[195] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992.

[196] H. Wu, C. Wang, Y. Tyshetskiy, A. Docherty, K. Lu, and L. Zhu. Adversarial examples for graph data: Deep insights into attack and defense. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 4816–4823, 2019.

[197] W. Xie, X. Mu, R. K. Lee, F. Zhu, and E. Lim. Unsupervised user identity linkage via factoid embedding. In *IEEE International Conference on Data Mining, ICDM 2018, Singapore, November 17-20, 2018*, pages 1338–1343, 2018.

[198] H. Xu, Y. Ma, H. Liu, D. Deb, H. Liu, J. Tang, and A. K. Jain. Adversarial attacks and defenses in images, graphs and text: A review. *CoRR*, abs/1909.08072, 2019b.

[199] K. Xu, L. Wang, M. Yu, Y. Feng, Y. Song, Z. Wang, and D. Yu. Cross-lingual knowledge graph alignment via graph matching neural network. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 3156–3161, 2019d.

[200] X. Zang, Y. Xie, J. Chen, and B. Yuan. Graph universal adversarial attacks: A few bad actors ruin graph learning models. *CoRR*, abs/2002.04784, 2020.

[201] H. Zhang, T. Zheng, J. Gao, C. Miao, L. Su, Y. Li, and K. Ren. Data poisoning attack against knowledge graph embedding. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 4853–4859, 2019a.

[202] J. Zhang and P. S. Yu. Integrated anchor and social link predictions across social networks. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 2125–2132, 2015.

[203] J. Zhang, X. Kong, and P. S. Yu. Predicting social links for new users across aligned heterogeneous social networks. In *2013 IEEE 13th International Conference on Data Mining, ICDM 2013, Dallas, TX, USA, December 7-10, 2013*, pages 1289–1294, 2013.

[204] S. Zhang and H. Tong. Attributed network alignment: Problem definitions and fast solutions. *IEEE Trans. Knowl. Data Eng.*, 31(9):1680–1692, 2019.

[205] S. Zhang, H. Tong, J. Tang, J. Xu, and W. Fan. ineat: Incomplete network alignment. In *2017 IEEE International Conference on Data Mining, ICDM 2017, New Orleans, LA, USA, November 18-21, 2017*, pages 1189–1194, 2017.

[206] S. Zhang, H. Tong, J. Xu, Y. Hu, and R. Maciejewski. ORIGIN: non-rigid network alignment. In *2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, December 9-12, 2019*, pages 998–1007, 2019c.

[207] Y. Zhang, J. Tang, Z. Yang, J. Pei, and P. S. Yu. COSNET: connecting heterogeneous social networks with local and global consistency. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*, pages 1485–1494, 2015.

[208] V. W. Zheng, M. Sha, Y. Li, H. Yang, Y. Fang, Z. Zhang, K. Tan, and K. C. Chang. Heterogeneous embedding propagation for large-scale e-commerce user alignment. In *IEEE International Conference on Data Mining, ICDM 2018, Singapore, November 17-20, 2018*, pages 1434–1439, 2018.

[209] Z. Zhong, Y. Cao, M. Guo, and Z. Nie. Colink: An unsupervised framework for user identity linkage. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5714–5721, 2018.

[210] F. Zhou, Z. Wen, G. Trajcevski, K. Zhang, T. Zhong, and F. Liu. Disentangled network alignment with matching explainability. In *2019 IEEE Conference on Computer Communications, INFOCOM 2019, Paris, France, April 29 - May 2, 2019*, pages 1360–1368, 2019a.

[211] J. Zhou and J. Fan. Translink: User identity linkage across heterogeneous social networks via translating embeddings. In *2019 IEEE Conference on Computer Communications, INFOCOM 2019, Paris, France, April 29 - May 2, 2019*, pages 2116–2124, 2019.

[212] K. Zhou, T. P. Michalak, M. Waniek, T. Rahwan, and Y. Vorobeychik. Attacking similarity-based link prediction in social networks. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '19, Montreal, QC, Canada, May 13-17, 2019*, pages 305–313, 2019b.

[213] X. Zhou, X. Liang, X. Du, and J. Zhao. Structure based user identification across social networks. *IEEE Trans. Knowl. Data Eng.*, 30(6):1178–1191, 2018b.

[214] Q. Zhu, X. Zhou, J. Wu, J. Tan, and L. Guo. Neighborhood-aware attentional representation for multilingual knowledge graphs. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 1943–1949, 2019b.

[215] D. Zügner and S. Günnemann. Adversarial attacks on graph neural networks via meta learning. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019.

[216] D. Zügner, A. Akbarnejad, and S. Günnemann. Adversarial attacks on neural networks for graph data. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pages 2847–2856, 2018.

[217] D. Zügner, O. Borchert, A. Akbarnejad, and S. Guennemann. Adversarial attacks on graph neural networks: Perturbations and their patterns. *To appear in ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2020.

[218] J. Baek, W. Jeong, J. Jin, J. Yoon, and S. J. Hwang. Personalized subgraph federated learning. *CoRR*, abs/2206.10206, 2022. doi: 10.48550/arXiv.2206.10206. URL `https://doi.org/10.48550/arXiv.2206.10206`.

[219] H. C. Bayram and I. Rekik. A federated multigraph integration approach for connectional brain template learning. In T. F. Syeda-Mahmood, X. Li, A. Madabhushi, H. Greenspan, Q. Li, R. M. Leahy, B. Dong, and H. Wang, editors, *Multimodal Learning for Clinical Decision Support - 11th International Workshop, ML-CDS 2021, Held in Conjunction with MICCAI 2021, Strasbourg, France, October 1, 2021, Proceedings*, volume 13050 of *Lecture Notes in Computer Science*, pages 36–47. Springer, 2021. doi: 10.1007/978-3-030-89847-2_4. URL `https://doi.org/10.1007/978-3-030-89847-2\_4`.

[220] D. Caldarola, M. Mancini, F. Galasso, M. Ciccone, E. Rodolà, and B. Caputo. Cluster-driven graph federated learning over multiple domains. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2021, virtual, June 19-25, 2021*, pages 2749–2758. Computer Vision Foundation / IEEE, 2021. doi: 10.1109/CVPRW53098.2021.00309. URL `https://openaccess.thecvf.com/content/CVPR2021W/LLID/html/Caldarola\_Cluster-Driven\_Graph\_Federated\_Learning\_Over\_Multiple\_Domains\_CVPRW\_2021\_paper.html`.

[221] N. Carlini, F. Tramèr, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. B. Brown, D. Song, Ú. Erlingsson, A. Oprea, and C. Raffel. Extracting training data from large language models. In M. Bailey and R. Greenstadt, editors, *30th USENIX Security Symposium, USENIX Security 2021, August 11-13, 2021*, pages 2633–2650. USENIX Association, 2021. URL `https://www.usenix.org/conference/usenixsecurity21/presentation/carlini-extracting`.

[222] S. Chakrabarti, H. Singh, S. Lohiya, P. Jain, and Mausam. Joint completion and alignment of multilingual knowledge graphs. In Y. Goldberg, Z. Kozareva, and Y. Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 11922–11938. Association for Computational Linguistics, 2022. URL `https://aclanthology.org/2022.emnlp-main.817`.

[223] C. Chen, W. Hu, Z. Xu, and Z. Zheng. Fedgl: Federated graph learning framework with global self-supervision. *CoRR*, abs/2105.03170, 2021.

[224] C. Chen, J. Zhou, L. Zheng, H. Wu, L. Lyu, J. Wu, B. Wu, Z. Liu, L. Wang, and X. Zheng. Vertically federated graph neural network for privacy-preserving node classification. In L. D. Raedt, editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 1959–1965. ijcai.org, 2022a. doi: 10.24963/ijcai.2022/272. URL `https://doi.org/10.24963/ijcai.2022/272`.

[225] F. Chen, G. Long, Z. Wu, T. Zhou, and J. Jiang. Personalized federated learning with a graph. In L. D. Raedt, editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 2575–2582. ijcai.org, 2022b. doi: 10.24963/ijcai.2022/357. URL `https://doi.org/10.24963/ijcai.2022/357`.

[226] L. Chen, Y. Zhao, B. Lyu, L. Jin, Z. Chen, S. Zhu, and K. Yu. Neural graph matching networks for chinese short text matching. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6152–6158, 2020a.

[227] M. Chen, W. Zhang, Z. Yao, X. Chen, M. Ding, F. Huang, and H. Chen. Meta-learning based knowledge extrapolation for knowledge graphs in the federated setting. In L. D. Raedt, editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 1966–1972. ijcai.org, 2022c. doi: 10.24963/ijcai.2022/273. URL `https://doi.org/10.24963/ijcai.2022/273`.

[228] Z. Fan, C. Mao, Y. Wu, and J. Xu. Spectral graph matching and regularized quadratic relaxations: Algorithm and theory. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, pages 2985–2995, 2020.

[229] M. Farid, W. J. Leong, and M. A. Hassan. A new two-step gradient-type method for large-scale unconstrained optimization. *Comput. Math. Appl.*, 59(10):3301–3307, 2010.

[230] Q. Gao, F. Wang, N. Xue, J. Yu, and G. Xia. Deep graph matching under quadratic constraint. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 5069–5078. Computer Vision Foundation / IEEE, 2021. doi: 10.1109/CVPR46437.2021.00503. URL `https://openaccess.thecvf.com/content/CVPR2021/html/Gao\_Deep\_Graph\_Matching\_Under\_Quadratic\_Constraint\_CVPR\_2021\_paper.html`.

[231] L. Guo, Q. Zhang, Z. Sun, M. Chen, W. Hu, and H. Chen. Understanding and improving knowledge graph embedding for entity alignment. In K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvári, G. Niu, and S. Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 8145–8156. PMLR, 2022. URL `https://proceedings.mlr.press/v162/guo22i.html`.

[232] S. Haller, L. Feineis, L. Hutschenreiter, F. Bernard, C. Rother, D. Kainmüller, P. Swoboda, and B. Savchynskyy. A comparative study of graph matching algorithms in computer vision. In S. Avidan, G. J. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, editors, *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XXIII*, volume 13683 of *Lecture Notes in Computer Science*, pages 636–653. Springer, 2022. doi: 10.1007/978-3-031-20050-2_37. URL `https://doi.org/10.1007/978-3-031-20050-2\_37`.

[233] C. He, K. Balasubramanian, E. Ceyani, Y. Rong, P. Zhao, J. Huang, M. Annavaram, and S. Avestimehr. Fedgraphnn: A federated learning system and benchmark for graph neural networks. *CoRR*, abs/2104.07145, 2021a. URL `https://arxiv.org/abs/2104.07145`.

[234] C. He, K. Balasubramanian, E. Ceyani, Y. Rong, P. Zhao, J. Huang, M. Annavaram, and S. Avestimehr. Fedgraphnn: A federated learning system and benchmark for graph neural networks. *CoRR*, abs/2104.07145, 2021b.

[235] C. He, E. Ceyani, K. Balasubramanian, M. Annavaram, and S. Avestimehr. Spreadgnn: Serverless multi-task federated learning for graph neural networks. *CoRR*, abs/2106.02743, 2021c.

[236] C. He, E. Ceyani, K. Balasubramanian, M. Annavaram, and S. Avestimehr. Spreadgnn: Decentralized multi-task federated learning for graph neural networks on molecular data. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 6865–6873. AAAI Press, 2022. URL `https://ojs.aaai.org/index.php/AAAI/article/view/20643`.

[237] Y. Hu, W. Liang, R. Wu, K. Xiao, W. Wang, X. Li, J. Liu, and Z. Qin. Quantifying and defending against privacy threats on federated knowledge graph embedding. In Y. Ding, J. Tang, J. F. Sequeda, L. Aroyo, C. Castillo, and G. Houben, editors, *Proceedings of the ACM Web Conference 2023, WWW 2023, Austin, TX, USA, 30 April 2023 - 4 May 2023*, pages 2306–2317. ACM, 2023. doi: 10.1145/3543507.3583450. URL `https://doi.org/10.1145/3543507.3583450`.

[238] J. J. E. Dennis and H. Wolkowicz. Sizing and least-change secant methods. *SIAM Journal on Numerical Analysis*, 30(5):1291–1314, 1993.

[239] E. Jiang and O. O. Koyejo. Weakly-supervised domain adaptation in federated learning. In *ICLR 2023 Submission*, 2023.

[240] M. Jiang, T. Jung, R. Karl, and T. Zhao. Federated dynamic GNN with secure aggregation. *CoRR*, abs/2009.07351, 2020.

[241] Z. Jiang, H. Rahmani, P. P. Angelov, S. Black, and B. M. Williams. Graph-context attention networks for size-varied deep graph matching. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 2333–2342. IEEE, 2022. doi: 10.1109/CVPR52688.2022.00238. URL `https://doi.org/10.1109/CVPR52688.2022.00238`.

[242] J. Jin, Z. T. Ke, and S. Luo. Network global testing by counting graphlets. In J. G. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 2338–2346. PMLR, 2018.

[243] H. Kang, Z. Li, and Q. Zhang. Communicational and computational efficient federated domain adaptation. *IEEE Trans. Parallel Distributed Syst.*, 33(12):3678–3689, 2022. doi: 10.1109/TPDS.2022.3167457. URL `https://doi.org/10.1109/TPDS.2022.3167457`.

[244] S. P. Karimireddy, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich, and A. T. Suresh. SCAFFOLD: stochastic controlled averaging for federated learning. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 5132–5143. PMLR, 2020.

[245] S. P. Karimireddy, M. Jaggi, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich, and A. T. Suresh. Mime: Mimicking centralized stochastic algorithms in federated learning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*, 2021.

[246] C. Ke and J. Honorio. Federated myopic community detection with one-shot communication. *CoRR*, abs/2106.07255, 2021.

[247] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

[248] A. Lalitha, O. C. Kilinc, T. Javidi, and F. Koushanfar. Peer-to-peer federated learning on graphs. *CoRR*, abs/1901.11173, 2019. URL `http://arxiv.org/abs/1901.11173`.

[249] W. Li, X. Liu, and Y. Yuan. SIGMA: semantic-complete graph matching for domain adaptive object detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 5281–5290. IEEE, 2022. doi: 10.1109/CVPR52688.2022.00522. URL `https://doi.org/10.1109/CVPR52688.2022.00522`.

[250] X. Li, W. Zhang, R.-H. Li, Y. Zhao, Y. Zhu, and G. Wang. A new paradigm for federated structure non-iid subgraph learning. In *ICLR 2023 Submission*, 2023.

[251] C. Liu, S. Zhang, X. Yang, and J. Yan. Self-supervised learning of visual graph matching. In S. Avidan, G. J. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, editors, *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XXIII*, volume 13683 of *Lecture Notes in Computer Science*, pages 370–388. Springer, 2022a. doi: 10.1007/978-3-031-20050-2_22. URL `https://doi.org/10.1007/978-3-031-20050-2\_22`.

[252] L. Liu, M. C. Hughes, S. Hassoun, and L. Liu. Stochastic iterative graph matching. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 6815–6825. PMLR, 2021. URL `http://proceedings.mlr.press/v139/liu21i.html`.

[253] W. Liu, L. Chen, Y. Chen, and W. Zhang. Accelerating federated learning via momentum gradient descent. *IEEE Trans. Parallel Distributed Syst.*, 31(8):1754–1766, 2020.

[254] X. Liu, H. Hong, X. Wang, Z. Chen, E. Kharlamov, Y. Dong, and J. Tang. Selfkg: Self-supervised entity alignment in knowledge graphs. In F. Laforest, R. Troncy, E. Simperl, D. Agarwal, A. Gionis, I. Herman, and L. Médini, editors, *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*, pages 860–870. ACM, 2022b. doi: 10.1145/3485447.3511945. URL `https://doi.org/10.1145/3485447.3511945`.

[255] G. Lyu, Y. Wu, and S. Feng. Deep graph matching for partial label learning. In L. D. Raedt, editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 3306–3312. ijcai.org, 2022. doi: 10.24963/ijcai.2022/459. URL `https://doi.org/10.24963/ijcai.2022/459`.

[256] G. Mei, Z. Guo, S. Liu, and L. Pan. SGNN: A graph neural network based federated learning approach by hiding structure. In *2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, December 9-12, 2019*, pages 2560–2568, 2019.

[257] C. Meng, S. Rambhatla, and Y. Liu. Cross-node federated graph neural network for spatio-temporal data modeling. In F. Zhu, B. C. Ooi, and C. Miao, editors, *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*, pages 1202–1211. ACM, 2021. doi: 10.1145/3447548.3467371. URL `https://doi.org/10.1145/3447548.3467371`.

[258] L. Meng, Y. Ren, J. Zhang, F. Ye, and P. S. Yu. Deep heterogeneous social network alignment. In *2019 IEEE First International Conference on Cognitive Machine Intelligence (CogMI), Los Angeles, CA, USA, December 12-14, 2019*, pages 43–52, 2019.

[259] A. Mitra, R. Jaafar, G. Pappas, and H. Hassani. Linear convergence in federated learning: Tackling client heterogeneity and sparse gradients. In *The 35th Conference on Neural Information Processing Systems, (NeurIPS'21)*, Online, December 6-14 2021.

[260] X. Ni, X. Xu, L. Lyu, C. Meng, and W. Wang. A vertical federated learning framework for graph convolutional network. *CoRR*, abs/2106.11593, 2021. URL `https://arxiv.org/abs/2106.11593`.

[261] K. Paramonov, D. Shemetov, and J. Sharpnack. Estimating graphlet statistics via lifting. In A. Teredesai, V. Kumar, Y. Li, R. Rosales, E. Terzi, and G. Karypis, editors, *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pages 587–595. ACM, 2019.

[262] J. Park, C. Tran, W. Shin, and X. Cao. Gradalign+: Empowering gradual network alignment using attribute augmentation. In M. A. Hasan and L. Xiong, editors, *Proceedings of the 31st ACM International Conference on Information & Knowledge Management, Atlanta, GA, USA, October 17-21, 2022*, pages 4374–4378. ACM, 2022. doi: 10.1145/3511808.3557605. URL https://doi.org/10.1145/3511808.3557605.

[263] S. Pei, L. Yu, G. Yu, and X. Zhang. Graph alignment with noisy supervision. In F. Laforest, R. Troncy, E. Simperl, D. Agarwal, A. Gionis, I. Herman, and L. Médini, editors, *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*, pages 1104–1114. ACM, 2022. doi: 10.1145/3485447.3512089. URL https://doi.org/10.1145/3485447.3512089.

[264] H. Peng, H. Li, Y. Song, V. W. Zheng, and J. Li. Differentially private federated knowledge graphs embedding. In G. Demartini, G. Zuccon, J. S. Culpepper, Z. Huang, and H. Tong, editors, *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021*, pages 1416–1425. ACM, 2021. doi: 10.1145/3459637.3482252. URL https://doi.org/10.1145/3459637.3482252.

[265] X. Peng, Z. Huang, Y. Zhu, and K. Saenko. Federated adversarial domain adaptation. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=HJezF3VYPB.

[266] Y. Qiao, Y. Wu, F. Duo, W. Lin, and J. Yang. Siamese neural networks for user identity linkage through web browsing. *IEEE Trans. Neural Networks Learn. Syst.*, 31 (8):2741–2751, 2020.

[267] J. Qu, H. Ling, C. Zhang, X. Lyu, and Z. Tang. Adaptive edge attention for graph matching with outliers. In Z. Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pages 966–972. ijcai.org, 2021. doi: 10.24963/ijcai.2021/134. URL https://doi.org/10.24963/ijcai.2021/134.

[268] L. Qu, N. Tang, R. Zheng, Q. V. H. Nguyen, Z. Huang, Y. Shi, and H. Yin. Semi-decentralized federated ego graph learning for recommendation. In Y. Ding, J. Tang, J. F. Sequeda, L. Aroyo, C. Castillo, and G. Houben, editors, *Proceedings of the ACM Web Conference 2023, WWW 2023, Austin, TX, USA, 30 April 2023 - 4 May 2023*, pages 339–348. ACM, 2023. doi: 10.1145/3543507.3583337. URL https://doi.org/10.1145/3543507.3583337.

[269] M. Z. Rácz and A. Sridhar. Correlated stochastic block models: Exact graph matching with applications to recovering communities. In M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 22259–22273, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/baf4f1a5938b8d520b328c13b51ccf11-Abstract.html.

[270] D. Randall. Efficient generation of random nonsingular matrices. *Random Struct. Algorithms*, 4(1):111–118, 1993.

[271] S. J. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečný, S. Kumar, and H. B. McMahan. Adaptive federated optimization. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*, 2021.

[272] Q. Ren, Q. Bao, R. Wang, and J. Yan. Appearance and structure aware robust deep visual graph matching: Attack, defense and beyond. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 15242–15251. IEEE, 2022. doi: 10.1109/CVPR52688.2022.01483. URL `https://doi.org/10.1109/CVPR52688.2022.01483`.

[273] E. Rizk and A. H. Sayed. A graph federated architecture with privacy preserving learning. In *22nd IEEE International Workshop on Signal Processing Advances in Wireless Communications, SPAWC 2021, Lucca, Italy, September 27-30, 2021*, pages 131–135. IEEE, 2021. doi: 10.1109/SPAWC51858.2021.9593148. URL `https://doi.org/10.1109/SPAWC51858.2021.9593148`.

[274] N. Shervashidze, S. V. N. Vishwanathan, T. Petri, K. Mehlhorn, and K. M. Borgwardt. Efficient graphlet kernels for large graph comparison. In D. A. V. Dyk and M. Welling, editors, *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics, AISTATS 2009, Clearwater Beach, Florida, USA, April 16-18, 2009*, volume 5 of *JMLR Proceedings*, pages 488–495. JMLR.org, 2009.

[275] H. A. Soufiani and E. M. Airoldi. Graphlet decomposition of a weighted network. In N. D. Lawrence and M. A. Girolami, editors, *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2012, La Palma, Canary Islands, Spain, April 21-23, 2012*, volume 22 of *JMLR Proceedings*, pages 54–63. JMLR.org, 2012.

[276] Y. Sun, N. S. T. Chong, and H. Ochiai. Multi-source domain adaptation based on federated knowledge alignment. *CoRR*, abs/2203.11635, 2022. doi: 10.48550/arXiv.2203.11635. URL `https://doi.org/10.48550/arXiv.2203.11635`.

[277] T. Suzumura, Y. Zhou, N. Barcardo, G. Ye, K. Houck, R. Kawahara, A. Anwar, L. L. Stavarache, D. Klyashtorny, H. Ludwig, and K. Bhaskaran. Towards federated graph learning for collaborative financial crimes detection. *CoRR*, abs/1909.12946, 2019.

[278] Y. Tan, Y. Liu, G. Long, J. Jiang, Q. Lu, and C. Zhang. Federated learning on non-iid graphs via structural knowledge sharing. *CoRR*, abs/2211.13009, 2022. doi: 10.48550/arXiv.2211.13009. URL `https://doi.org/10.48550/arXiv.2211.13009`.

[279] Z. Tian, Y. Ding, R. Zhang, J. Liu, and K. Ren. Towards federated learning of deep graph neural networks. In *ICLR 2023 Submission*, 2023.

[280] B. Wang, A. Li, H. Li, and Y. Chen. Graphfl: A federated learning framework for semi-supervised node classification on graphs. *CoRR*, abs/2012.04187, 2020a.

[281] B. Wang, A. Li, M. Pang, H. Li, and Y. Chen. Graphfl: A federated learning framework for semi-supervised node classification on graphs. In X. Zhu, S. Ranka, M. T. Thai, T. Washio, and X. Wu, editors, *IEEE International Conference on Data Mining, ICDM 2022, Orlando, FL, USA, November 28 - Dec. 1, 2022*, pages 498–507. IEEE, 2022a. doi: 10.1109/ICDM54844.2022.00060. URL `https://doi.org/10.1109/ICDM54844.2022.00060`.

[282] C. Wang, Y. Wang, Z. Zhao, D. Qin, X. Luo, and T. Qin. Credible seed identification for large-scale structural network alignment. *Data Min. Knowl. Discov.*, 34(6):1744–1776, 2020b.

[283] C. Wang, B. Chen, G. Li, and H. Wang. FL-AGCNS: federated learning framework for automatic graph convolutional network search. *CoRR*, abs/2104.04141, 2021a.

[284] F. Wang, N. Xue, J. Yu, and G. Xia. Zero-assignment constraint for graph matching with outliers. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 3030–3039, 2020c.

[285] J. Wang, Z. Xu, Z. Garrett, Z. Charles, L. Liu, and G. Joshi. Local adaptivity in federated learning: Convergence and consistency. In *The International Workshop on Federated Learning for User Privacy and Data Confidentiality in Conjunction with ICML 2021, (FL-ICML'21)*, Online, December 6-14 2021b.

[286] T. Wang, H. Liu, Y. Li, Y. Jin, X. Hou, and H. Ling. Learning combinatorial solver for graph matching. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 7565–7574, 2020e.

[287] Z. Wang, W. Kuang, Y. Xie, L. Yao, Y. Li, B. Ding, and J. Zhou. Federatedscope-gnn: Towards a unified, comprehensive and efficient package for federated graph learning. In A. Zhang and H. Rangwala, editors, *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18, 2022*, pages 4110–4120. ACM, 2022b. doi: 10.1145/3534678.3539112. URL `https://doi.org/10.1145/3534678.3539112`.

[288] D. A. Weighill, M. B. Guebila, C. M. Lopes-Ramos, K. Glass, J. Quackenbush, J. Platig, and R. Burkholz. Gene regulatory network inference as relaxed graph matching. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 10263–10272. AAAI Press, 2021. URL `https://ojs.aaai.org/index.php/AAAI/article/view/17230`.

[289] C. Wu, F. Wu, Y. Cao, Y. Huang, and X. Xie. Fedgnn: Federated graph neural network for privacy-preserving recommendation. In *The International Workshop on Federated Learning for User Privacy and Data Confidentiality in Conjunction with ICML 2021, (FL-ICML'21)*, Online, December 6-14 2021.

[290] Y. xiang Yuan. A modified bfgs algorithm for unconstrained optimization. *IMA Journal of Numerical Analysis*, 11(3):325–332, 1991.

[291] H. Xie, J. Ma, L. Xiong, and C. Yang. Federated graph classification over non-iid graphs. In M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 18839–18852, 2021. URL `https://proceedings.neurips.cc/paper/2021/hash/9c6947bd95ae487c81d4e19d3ed8cd6f-Abstract.html`.

[292] H. Xie, L. Xiong, and C. Yang. Federated node classification over graphs with latent link-type heterogeneity. In Y. Ding, J. Tang, J. F. Sequeda, L. Aroyo, C. Castillo, and G. Houben, editors, *Proceedings of the ACM Web Conference 2023, WWW 2023, Austin, TX, USA, 30 April 2023 - 4 May 2023*, pages 556–566. ACM, 2023. doi: 10.1145/3543507.3583471. URL `https://doi.org/10.1145/3543507.3583471`.

[293] K. Xin, Z. Sun, W. Hua, W. Hu, J. Qu, and X. Zhou. Large-scale entity alignment via knowledge graph merging, partitioning and embedding. In M. A. Hasan and L. Xiong, editors, *Proceedings of the 31st ACM International Conference on Information & Knowledge Management, Atlanta, GA, USA, October 17-21, 2022*, pages 2240–2249. ACM, 2022. doi: 10.1145/3511808.3557374. URL `https://doi.org/10.1145/3511808.3557374`.

[294] X. Yang, Z. Liu, and H. Qiaoxu. Incorporating discrete constraints into random walk-based graph matching. *IEEE Trans. Syst. Man Cybern. Syst.*, 50(4):1406–1416, 2020.

[295] Z. Ye, T. Yenamandra, F. Bernard, and D. Cremers. Joint deep multi-graph matching and 3d geometry learning from inhomogeneous 2d image collections. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 3125–3133. AAAI Press, 2022. URL `https://ojs.aaai.org/index.php/AAAI/article/view/20220`.

[296] L. Yu, J. Xu, and X. Lin. Seedgnn: Graph neural networks for supervised seeded graph matching. *CoRR*, abs/2205.13679, 2022. doi: 10.48550/arXiv.2205.13679. URL `https://doi.org/10.48550/arXiv.2205.13679`.

[297] T. Yu, R. Wang, J. Yan, and B. Li. Deep latent graph matching. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 12187–12197. PMLR, 2021. URL `http://proceedings.mlr.press/v139/yu21d.html`.

[298] H. Zhang, T. Shen, F. Wu, M. Yin, H. Yang, and C. Wu. Federated graph learning - A position paper. *CoRR*, abs/2105.11099, 2021a.

[299] K. Zhang, C. Yang, X. Li, L. Sun, and S. Yiu. Subgraph federated learning with missing neighbor generation. In M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 6671–6682, 2021b. URL `https://proceedings.neurips.cc/paper/2021/hash/34adeb8e3242824038aa65460a47c29e-Abstract.html`.

[300] K. Zhang, Y. Wang, H. Wang, L. Huang, C. Yang, X. Chen, and L. Sun. Efficient federated learning on knowledge graphs via privacy-preserving relation embedding aggregation. In Y. Goldberg, Z. Kozareva, and Y. Zhang, editors, *Findings of the Association for Computational Linguistics: EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 613–621. Association for Computational Linguistics, 2022. URL `https://aclanthology.org/2022.findings-emnlp.43`.

[301] S. Zhang and H. Tong. Network alignment: Recent advances and future directions. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pages 3521–3522, 2020.

[302] S. Zhang, H. Tong, Y. Xia, L. Xiong, and J. Xu. Nettrans: Neural cross-network transformation. In R. Gupta, Y. Liu, J. Tang, and B. A. Prakash, editors, *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 986–996. ACM, 2020. doi: 10.1145/3394486.3403141. URL `https://doi.org/10.1145/3394486.3403141`.

[303] S. Zhang, H. Tong, L. Jin, Y. Xia, and Y. Guo. Balancing consistency and disparity in network alignment. In F. Zhu, B. C. Ooi, and C. Miao, editors, *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*, pages 2212–2222. ACM, 2021c. doi: 10.1145/3447548.3467331. URL `https://doi.org/10.1145/3447548.3467331`.

[304] X. Zhang, M. Hong, and J. Chen. GLASU: A communication-efficient algorithm for federated learning with vertically distributed graph data. *CoRR*, abs/2303.09531, 2023. doi: 10.48550/arXiv.2303.09531. URL `https://doi.org/10.48550/arXiv.2303.09531`.

[305] K. Zhao, S. Tu, and L. Xu. IA-GM: A deep bidirectional learning method for graph matching. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 3474–3482. AAAI Press, 2021. URL `https://ojs.aaai.org/index.php/AAAI/article/view/16461`.

[306] D. Zheng, M. Wang, Q. Gan, Z. Zhang, and G. Karypis. Learning graph neural networks with deep graph library. In A. E. F. Seghrouchni, G. Sukthankar, T. Liu, and M. van Steen, editors, *Companion of The 2020 Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 305–306. ACM / IW3C2, 2020.

[307] F. Zhou, C. Cao, G. Trajcevski, K. Zhang, T. Zhong, and J. Geng. Fast network alignment via graph meta-learning. In *39th IEEE Conference on Computer Communications, INFOCOM 2020, Toronto, ON, Canada, July 6-9, 2020*, pages 686–695, 2020a.

[308] J. Zhou, C. Chen, L. Zheng, H. Wu, J. Wu, X. Zheng, B. Wu, Z. Liu, and L. Wang. Vertically federated graph neural network for privacy-preserving node classification. *CoRR*, abs/2005.11903, 2020b.

[309] T. Zhou, E. Lim, R. K. Lee, F. Zhu, and J. Cao. Retrofitting embeddings for unsupervised user identity linkage. In *Advances in Knowledge Discovery and Data Mining - 24th Pacific-Asia Conference, PAKDD 2020, Singapore, May 11-14, 2020, Proceedings, Part I*, pages 385–397, 2020c.

[310] J. Zhu, D. Koutra, and M. Heimann. CAPER: coarsen, align, project, refine - A general multilevel framework for network alignment. In M. A. Hasan and L. Xiong, editors, *Proceedings of the 31st ACM International Conference on Information & Knowledge Management, Atlanta, GA, USA, October 17-21, 2022*, pages 4747–4751. ACM, 2022a. doi: 10.1145/3511808.3557563. URL `https://doi.org/10.1145/3511808.3557563`.

[311] R. Zhu, X. Luo, M. Ma, and P. Wang. Adaptive graph convolutional network for knowledge graph entity alignment. In Y. Goldberg, Z. Kozareva, and Y. Zhang, editors, *Findings of the Association for Computational Linguistics: EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 6011–6021. Association for Computational Linguistics, 2022b. URL `https://aclanthology.org/2022.findings-emnlp.444`.

[312] X. Zhu, G. Li, and W. Hu. Heterogeneous federated knowledge graph embedding learning and unlearning. In Y. Ding, J. Tang, J. F. Sequeda, L. Aroyo, C. Castillo, and G. Houben, editors, *Proceedings of the ACM Web Conference 2023, WWW 2023, Austin, TX, USA, 30 April 2023 - 4 May 2023*, pages 2444–2454. ACM, 2023. doi: 10.1145/3543507.3583305. URL `https://doi.org/10.1145/3543507.3583305`.

[313] M. Zitnik and J. Leskovec. Predicting multicellular function through multi-layer tissue networks. *Bioinform.*, 33(14):i190–i198, 2017.

[314] Ian J. Goodfellow and Jonathon Shlens and Christian Szegedy Explaining and Harnessing Adversarial Examples *ICLR*, 2015.

[315] Guoming Zhang and Chen Yan and Xiaoyu Ji andTaimin Zhang and Tianchen Zhang and Wenyuan Xu DolphinAtack: Inaudible Voice Commands *CoRR*, 2017.

[316] John X. Morris and Eli Lifland and Jin Yong Yoo and Jake Grigsby and Di Jin and Yanjun Qi TextAttack: A Framework for Adversarial Attacks, Data Augmentation, and Adversarial Training in NLP *EMNLP*, 2020.