Appearance and Structure Aware Robust Deep Visual Graph Matching: Attack, Defense and Beyond

Qibing Ren, Qingquan Bao, Runzhong Wang, Junchi Yan*
Department of CSE & MoE Key Lab of Artifical Intelligence, Shanghai Jiao Tong University
{renqibing,faust-bqq,runzhong.wang,yanjunchi}@sjtu.edu.cn

Abstract

Despite the recent breakthrough of high accuracy deep graph matching (GM) over visual images, the robustness of deep GM models is rarely studied which yet has been revealed an important issue in modern deep nets, ranging from image recognition to graph learning tasks. We first show that an adversarial attack on keypoint localities and the hidden graphs can cause significant accuracy drop to deep GM models. Accordingly, we propose our defense strategy, namely Appearance and Structure Aware Robust Graph Matching (ASAR-GM). Specifically, orthogonal to de facto adversarial training (AT), we devise the Appearance Aware Regularizer (AAR) on those appearance-similar keypoints between graphs that are likely to confuse. Experimental results show that our ASAR-GM achieves better robustness compared to AT. Moreover, our locality attack can serve as a data augmentation technique, which boosts the state-of-the-art GM models even on the clean test dataset.

1. Introduction

Graph matching (GM), as one of the most important research topics in the graph domain with wide applications in vision and pattern recognition, aims to find node-to-node correspondence among graphs. The matching of visual graphs has been intensively studied over the decades, such as image keypoint matching [42], scene graph discovery [6], and vision-text retrieval [46], especially since the recent advances in combining deep neural networks and (visual) GM [51]. Despite the success of deep GM, deep neural networks (DNN)s are found vulnerable to small input perturbations which are imperceptible to humans [3, 38]. For example, in image classification, carefully designed small perturbations on image pixels can fool neural classifiers [18], and in graph domain, attackers perturb graph structures and its attributes to cause failures of graph learning tasks such as node classification [10, 13, 37, 40, 59], community detection [7, 27], link prediction [32], etc. However, there is little work which considers the vulnerability of deep GM for vision, or more specifically matching image keypoints, which is recently a trending research topic [16, 17, 23, 24, 34, 42, 44, 48, 49, 57]. Since noise can be easily injected into images and graphs, a natural question that we would answer in our work is: How to design an effective adversarial attack on GM, perturbing images and their hidden graphs simultaneously?

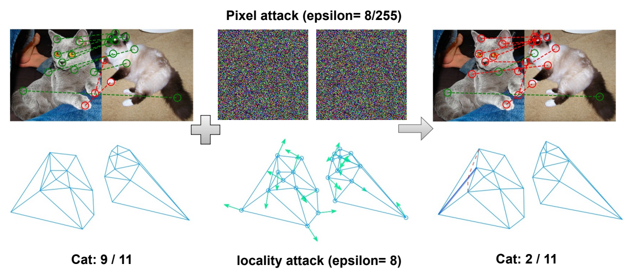


Figure 1. The proposed imperceptible adversarial attacks. **Left**: input paired images and their graphs; **Middle**: adversarial perturbations on the pixel and locality; **Right**: induced adversarial data. After being attacked, the appearance around the keypoints remains unchanged while the graph structure get perturbed: the red dotted line means the edge on the original graph being removed while the blue real line denotes the added edge on the new perturbed graph.

In the context of visual GM, it is natural for the attacker to consider perturbing image pixels, which are directly related to node features of visual graphs. Besides, deep GM also takes the keypoint locality and the induced hidden graphs as input. However, such way of adding or removing edges of the hidden graphs, as a common graph attack baseline [33, 56], is NOT feasible to visual GM: with the annotated keypoint locality, graph structure of GM is determined by certain domain knowledge, e.g., Delaunay triangulation [11], such that any operation on graph edges can be easily recovered. We instead focus on the location of keypoints. Among the common deep GM pipelines [44, 51], the location property of keypoints in each graph is crucial to the final matching performance since it affects how features of keypoints are extracted from the whole image through bi-

*Correspondence author. This work was in part supported by China Key Research and Development Program (2020AAA0107600), Shanghai Municipal Science and Technology Major Project (2021SHZDZX0102).

Table 1. Comparisons of the attack and defense of vision and graph tasks. The column "Attack Object" denotes the specific perturbation object about the input; "Attack Type" denotes which way to perturb the input; the column "Similarity Metric" shows how to measure the similarity between clean and adversarial example, and "Defense Objective Function" is the minimization goal of the defender.

Task	Attack Object	Attack Type	Similarity Measure	Defense Objective Function
image classification [18]	image	flip pixels	l_p norm	cross-entropy
object tracking [22]	image sequence	flip pixels	l_p norm	cross-entropy with smooth L1
node classification [10, 37]	graph	inject nodes; add/delete edges	ratio of perturbed nodes(edges)	cross-entropy
graph matching [33, 56]	graph	add/delete edges	ratio of perturbed edges	pairwise cosine similarity
visual graph matching (ours)	visual graph	flip pixels; perturb keypoint locality	l_p norm	binary cross-entropy

linear interpolation and directly determines the graph structure derived by Delaunay triangulation. However, the location of keypoints suffers from its inherent instability due to the randomness of human labeling or keypoint detectors, which means small yet malicious noise could be easily added without being detected. Therefore, we propose to perturb keypoint locality as an effective adversarial attack.

Towards defending against adversarial attacks, adversarial training (AT) [30] has become a widely-recognized principled defense mechanism by training models on adversarial examples while it suffers from lowering accuracy on clean test examples. Moreover, for graph learning, there are efforts in improving robustness against adversarial attacks in node classification [47, 54, 58], graph classification [25], community detection [21], etc. However, those defense mechanisms only focus on single graph learning tasks while GM learns to analyze intersections among graphs such that these methods cannot be directly applied to GM. In this paper, based on our analysis of the vulnerability of GM, we explore a new way of defending against adversarial attacks.

Our defense mechanism derives from two insights. First, we show that that adversarial attacks tend to confuse keypoints with similar appearance and those appearance-similar keypoints usually occur in three cases: (1) shape similarity, e.g. the two ears of a cat; (2) texture similarity, e.g. the wither and tail of a cat; (3) structural symmetry, e.g. the four roof corners of a car. Such appearance-similarity depends on some prior in the dataset. For two graphs, if we can select these appearance-similar keypoints between them and explicitly enlarge their disparity in the probabilistic space of model outputs, the model robustness could get enhanced. Moreover, since our regularization strategy works in output space, which is orthogonal to AT that generates worst-case example in input space, we can further improve model robustness by combining them together.

To this end, we take the initiative on studying the robustness of visual GM. On the attacker's side, we propose an effective keypoint locality attack and combine it with pixel attack to devise an even stronger attack. For defense, we analyze the attack pattern and discover that those appearance-similar keypoints can be inferred from the result of our adversarial attack. Then we design a regularization term, namely **A**ppearance **A**ware **R**egularizer (AAR), to regularize the discrepancy of features of keypoints which share similar appearance in the low-dimensional embedding space. Finally, we propose our defense strategy, namely **A**ppearance and **S**tructure **A**ware **R**obust **G**raph **M**atching (ASAR-GM) on the basis of AT. **The highlights are:**

1) We analyze the vulnerability of deep (visual) graph matching (GM) under adversarial attacks and design an effective locality attack, which perturbs the keypoint locations and hidden graph structure together. Moreover, stronger adversarial data is generated by combining our locality attack and pixel attack together. Our work differs from two recent GM attack/defense works as they only focus on adding/deleting the edges without manipulating on visual images as also considered in our method (see Table 1).

2) We propose our defense strategy, namely **A**ppearance and **S**tructure **A**ware **R**obust **G**raph **M**atching (ASAR-GM) to enhance robustness. Specifically, we show that adversarial attacks tend to utilize appearance-similar keypoints among graphs to fool the matching of the model. As such, we design a regularization term: **A**ppearance **A**ware **R**egularizer (AAR), to enlarge the disparity among appearance-similar keypoints in graph. Our AAR can be naturally integrated into the framework of AT, which brings better clean accuracy and robustness.

3) Experiments on real-world benchmarks validate the effectiveness of our attack on various deep GM baselines [34, 42, 48] including the state-of-the-art NGMv2 [44]. Our attack also shows strong transferability in the black-box attack setting. For defense, ASAR-GM achieves better clean accuracy and robustness over defense baselines.

4) Last but importantly, while adversarial examples are often viewed a threat to DNN, our locality attack serves as a data augmentation to improve generalization ability of deep GM because perturbations on locality induce various graph structures for training, making our model a new GM SOTA.

2. Related Work

Deep Graph Matching. The pioneer work [31] considers graph alignment by embedding on individual graphs. With the remarkable performance of deep neural networks (DNNs) in vision, deep learning has been applied for GM on images since the proposal of the seminal work [51], which utilizes a convolutional neural network (CNN) to extract node features and builds an end-to-end model with spectral matching. Since then, deep (visual) GM has become a trending topic: [17, 23, 42, 43, 55, 57] introduce graph neural networks (GNN)s [35] to improve GM by encoding graph structural information; [48] proposes an edge embedding module and Hungarian-based attention mechanism; the work [34] proposes an end-to-end deep GM architecture combining unmodified combinatorial solvers with deep learning together; NGMv2 [44], as our main defense baseline model in Paper, deals with the general Lawler's QAP form [29], which solves GM via applying vertex classification on the association graph. By adopting more advanced feature extractors e.g. [15], NGMv2 achieves the state-of-the-art performance for deep GM.

Adversarial Attack & Defense on GM. [56] focuses on dealing with the raw graph data without vision information. It generates adversarial examples by maximizing a node density estimation function built by kernel density estimation (KDE) during a meta-learning-based PGD attack. They craft adversarial data by inserting/deleting edges. However, such an attack is NOT feasible for visual GM, because firstly, the construction of (visual) graphs is determined by certain domain knowledge, e.g. Delaunay triangulation; secondly, the perturbed edges are no longer "imperceptible" and could be detected and recovered easily. [33] enhances the robustness of traditional GM by penalizing the dense region of nodes against the node density attack [56], and detecting the adversarial examples from the inputs, which are different from our proactive defense, i.e., increasing robustness of the victim models against adversarial examples. This paper also differs from papers studying the robustness of object tracking [22,28], where only the visual features are considered. The recent work [50] enhances the robustness of visual GM against "natural" noise in images e.g. deformations, rotations, and outliers. But it dost not consider to defend against the designed adversarial attacks.

3. Preliminaries

3.1. Problem Definition

We mainly focus on the visual GM task: given an image pair $\mathbf{c}$=($\mathbf{c}^1, \mathbf{c}^2$), each of them is annotated with n keypoints, and their annotated keypoint locality set $\mathbf{z}$=($\mathbf{z}^1, \mathbf{z}^2$), where $\mathbf{z}^1, \mathbf{z}^2 \in \mathbb{R}^{n\times 2}$. Moreover, we treat the hidden keypoint graphs G=(G^1, G^2) as the general attribute graph, i.e., $G^1 = \{V^1, E^1, \mathbf{G}^1, \mathbf{H}^1\}$ and $G^2 = \{V^2, E^2, \mathbf{G}^2, \mathbf{H}^2\}$. Here V is the node set, E is the edge set and $|V^1|$=n, $|E^1|$=m_1,$|V^2|$=n, $|E^2|$=m_2. The connectivity of two graphs are represented by $\mathbf{G}^1$, $\mathbf{H}^1 \in \{0,1\}^{n\times m_1}$ and $\mathbf{G}^2$, $\mathbf{H}^2 \in \{0,1\}^{n\times m_2}$, where $\mathbf{G}_{i,k} = \mathbf{H}_{i,k} = 1$ means edge k links node i to node j and $\mathbf{A}^1 = \mathbf{G}^1\mathbf{H}^{1\top}$, $\mathbf{A}^2 = \mathbf{G}^2\mathbf{H}^{2\top}$ are the adjacency matrices of two graphs.

Graph matching. It can be written as quadratic assignment programming (QAP) [29], where $\mathbf{X} \in \mathbb{R}^{n\times n}$ is a permutation matrix for node-to-node correspondence[1], and vec($\mathbf{X}$) is its column-vectorized version:

$$\begin{aligned}&\max_{\mathbf{X}} J(\mathbf{X}) = \text{vec}(\mathbf{X})^\top \mathbf{K}\text{vec}(\mathbf{X}) \\ &s.t.\ \mathbf{X} \in \{0,1\}^{n\times n}, \mathbf{X}\mathbf{1}_n = \mathbf{1}_n, \mathbf{X}^\top\mathbf{1}_n = \mathbf{1}_n\end{aligned} \tag{1}$$

where $\mathbf{K} \in \mathbb{R}^{n^2\times n^2}$ is the affinity matrix whose diagonal and off-diagonal elements store the node-to-node and edge-to-edge affinities. The goal of GM is to maximize the objective $J(\mathbf{X})$ with the assumption that perfect matching corresponds to the highest affinity score.

Deep graph matching. To enable end-to-end learning, Lawler's QAP in Eq. 1 is relaxed via (partial) doubly-stochastic relaxation for $\mathbf{S}$ whose rows/columns sum to 1:

$$\begin{aligned}&\max_{\mathbf{S}} J(\mathbf{S}) = \text{vec}(\mathbf{S})^\top \mathbf{K}\text{vec}(\mathbf{S}) \\ &s.t.\ \mathbf{S} \in [0,1]^{n\times n}, \mathbf{S}\mathbf{1}_n = \mathbf{1}_n, \mathbf{S}^\top\mathbf{1}_n = \mathbf{1}_n\end{aligned} \tag{2}$$

Deep GM methods recently proposed deal with images with keypoints as inputs and solve such QAP problem in Eq. 2 in an end-to-end manner [34, 42–44, 48, 51, 55]. As Fig. 2 shows, these methods usually consist of three components: keypoint feature extractor, affinity learning, and final correspondence solver. Let f denote the CNN layers taking image pairs ($\mathbf{c}^1, \mathbf{c}^2$) for node (and edge) feature extraction, g denote the affinity learning layer for generating the affinity matrix $\mathbf{K}$, and the correspondence solver h for the final permutation. In this paper, we focus on the vulnerability of the current state-of-the-art model NGMv2 [44], where the matching problem is translated into a vertex classification task and binary cross-entropy (BCE) loss is utilized.

3.2. Adversarial Attack

For clarity, here we only consider adversarial attacks on image pixel. We denote the end-to-end deep GM pipeline as $\mathcal{M} : (\mathbf{c}^1, \mathbf{c}^2) \in [0,1]^D \mapsto \mathbf{X} \in \{0,1\}^{n\times n}$. Adversarial attack usually aims to find the worst-case example within a ball around the clean sample, $\mathcal{B}_\epsilon(\mathbf{c}) = \{\mathbf{c}' : d_p(\mathbf{c}, \mathbf{c}') \le \epsilon\}$, and $d_p(\mathbf{c}, \mathbf{c}')$= $\| \mathbf{c}' - \mathbf{c} \|_p$ is the similarity metric, where ℓ-∞ norm is chosen in our experiment.

White-box Attack. In the white-box setting, the attacker has the access to full information of models. Following Fast Gradient Sign Method (FGSM) [18] which adds perturbations along the direction of gradient descent, a popular and effective iterative gradient-based method, Projected Gradient Descent (PGD) attack [30], is proposed:

$$\mathbf{c}'_{k+1} = \Pi_{\mathcal{B}_\epsilon(\mathbf{c})}(\mathbf{c}'_k + \alpha sign(\nabla_{\mathbf{c}'_k} L(\mathcal{M}(\mathbf{c}'_k), y; \theta))) \tag{3}$$

where $\Pi_{\mathcal{B}_\epsilon(\mathbf{c})}(\cdot)$ is the projection function that projects the current adversarial example back to the ϵ-ball, L is the loss function and θ is model parameters.

Black-box Attack. The black-box attacker only knows outputs of the model. One type of black-box attacks is query-based methods: generating adversarial examples by querying the target model multiple times to perform random sampling [1, 19] or estimate gradients of the target model [20].

[1] We assume surjection between the nodes of two graphs as the most popular experiment setting for deep GM.

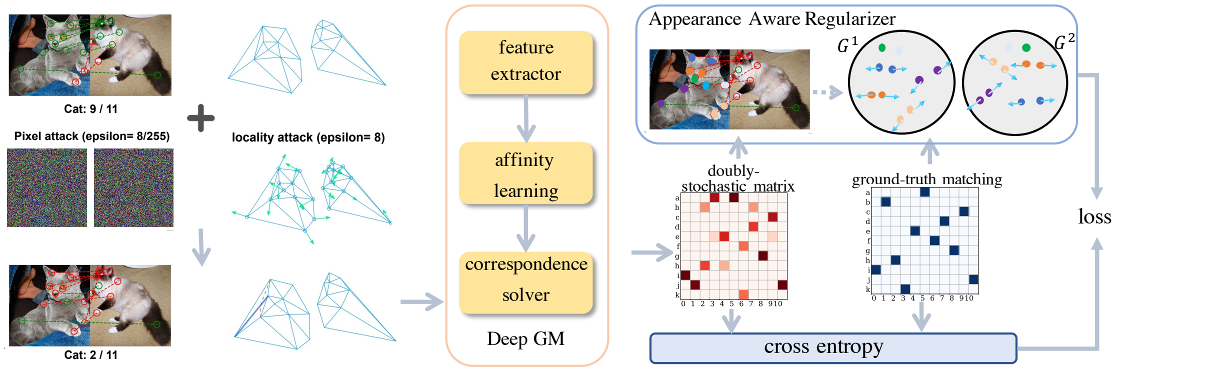


Figure 2. Pipeline of ASAR-GM: ASAR-GM receives our adversarial data as input and derives the predicted soft matching, i.e., doubly-stochastic matrix through i) feature extractor, ii) affinity learning layer, and iii) correspondence solver. Then ASAR-GM builds and trains on its Appearance Aware Regularizer: find the appearance-similar groups and enlarge their disparity in the embedding space.

The other popular attack is transfer-based: based on a surrogate model, the attacker either generates adversarial data and then transfer them to the target model or base on them to estimate gradients of loss of the target model [5, 8].

3.3. Adversarial Training

Towards the resistance of adversarial examples, adversarial training (AT) [30] trains models on adversarial examples instead of clean data. Specifically, the adversarial examples are generated by PGD attack in Eq. 3 and AT can be formulated as a bi-level optimization task:

$$\min_{\theta} E_{\mathbf{c},\mathbf{y}} \max_{\mathbf{c}' \in \mathcal{B}_\epsilon(\mathbf{c})} L(\mathcal{M}(\mathbf{c}'), \mathbf{y}; \theta) \tag{4}$$

Algorithm 1 Adversarial Attack with **V**isual and **S**tructural Manipulation (**VS-Attack**).

Input: A pair of images $\mathbf{c} = (\mathbf{c}^1, \mathbf{c}^2)$, its keypoint sets $\mathbf{z} = (\mathbf{z}^1, \mathbf{z}^2)$, and its two graphs (G^1, G^2); loss function L and model parameters θ; perturbation budget $(\epsilon_\mathbf{c}, \epsilon_\mathbf{z})$, perturbation steps m, and step size α; ground-truth matching $\mathbf{X}^{gt}$.
Output: Perturbed image $\mathbf{c}'$, keypoint $\mathbf{z}'$, and graph pair (G'^1, G'^2).
Initialize the adversarial example $\mathbf{c}'_0, \mathbf{z}'_0 \leftarrow \mathbf{c}, \mathbf{z}$.
for k in $(0, 1, \ldots, m-1)$ **do**
1. Calculate gradients: $\{g_{\mathbf{c}'_k}, g_{\mathbf{z}'_k}\} \leftarrow \{\nabla_{\mathbf{c}'_k} L(\mathbf{c}'_k, \mathbf{z}'_k, \mathbf{X}^{gt}; \theta), \nabla_{\mathbf{z}'_k} L(\mathbf{c}'_k, \mathbf{z}'_k, \mathbf{X}^{gt}; \theta)\}$.
2. Clip & Update pixel and locality: $\{\mathbf{c}'_{k+1}, \mathbf{z}'_{k+1}\} \leftarrow \{\Pi_{\mathcal{B}_\epsilon(\mathbf{c})}(\mathbf{c}'_k + \alpha sign(g_{\mathbf{c}'_k})), \Pi_{\mathcal{B}_\epsilon(\mathbf{z})}(\mathbf{z}'_k + \alpha sign(g_{\mathbf{z}'_k}))\}$ via Eq. 3.
3. Update graph: $(G'^1_k, G'^2_k) \leftarrow \mathbf{z}'_k$ by Delaunay triangulation.

end for

4. Imperceptible Adversarial Attack by Iterative Visual and Structural Manipulation

This section introduces a strong adversarial attack by iteratively updating visual and structural information of inputs. Sec. 4.1 analyzes the vulnerability of deep GM. While Sec. 4.2 gives our adversarial attack with details in Sec. 4.3.

4.1. Motivation

In Sec. 3.1, we introduce the common pipeline of deep two-graph matching as shown in Fig. 2: after building graphs by Delaunay triangulation [11], node features are obtained via a feature extractor f based on the keypoint locations and edge features are constituted based on node features and topology information of G^1, G^2, after which the affinity matrix $\mathbf{K}$ is initialized based on the node (edge) features. The initialized $\mathbf{K}$ is sent to the affinity learning layer g e.g. GNNs to learn the node-to-node and edge-to-edge similarity. Finally the predicted permutation matrix $\mathbf{X}$ gets obtained by the correspondence solver h.

We include the straight forward idea of attacking the image pixels. Besides, we can infer from the above pipeline that the location of keypoints $(\mathbf{z}^1, \mathbf{z}^2)$ affects how features of keypoints are extracted from the image and directly determines the graph structure derived by Delaunay triangulation. However, the location of keypoints suffers from its inherent instability due to the randomness of human labeling or keypoint detectors, which means small yet malicious noise could be easily added without being detected. Therefore, we also propose to perturb on keypoint locations.

4.2. Objective Design

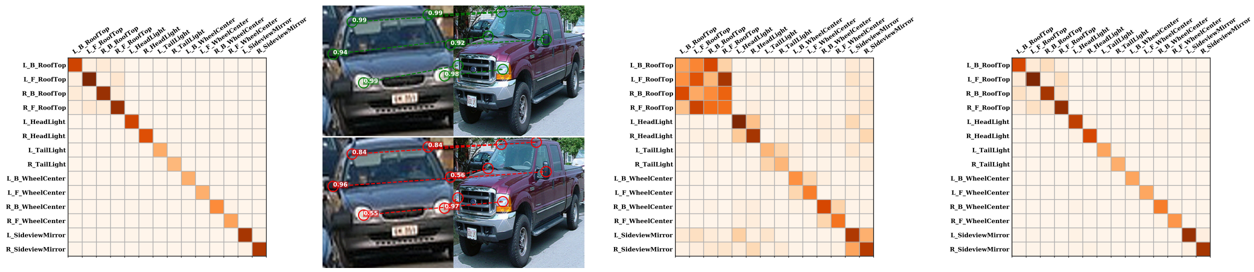
Given the analysis above, we explore a way of attacking both the image and graph through perturbing image pixel and keypoint locations simultaneously. We propose a joint optimization objective function as follows:

$$\begin{gathered} \max_{\mathbf{c}', \mathbf{z}'} \max_{G'} L(\mathbf{c}', \mathbf{z}', G', \mathbf{X}^{gt}; \theta) \\ s.t.\ d_\infty(\mathbf{c}', \mathbf{c}) \leq \epsilon_\mathbf{c} \quad d_\infty(\mathbf{z}', \mathbf{z}) \leq \epsilon_\mathbf{z} \end{gathered} \tag{5}$$

where $\mathbf{X}^{gt}$ is the ground-truth permutation; $\epsilon_\mathbf{c}$ and $\epsilon_\mathbf{z}$ are the perturbation budget to control how imperceptible the adversarial examples are to humans. Note that after perturbing the keypoint locality $\mathbf{z}'$, we further reconstruct the hidden graph G' based on Delaunay triangulation to boost the attack effectiveness. The pseudo code is given in Alg. 1.

4.3. Implementation

Since existing deep GM allows end-to-end learning, we can readily implement a PGD-like attack on pixels and keypoint locations by maximizing loss in Eq. 3. We denote



(a) baseline model without attacks (b) an attack example (c) baseline model under attacks (d) robust model under attacks

Figure 3. Analysis and visualization on keypoints of the "car" class for our assumption of appearance-similar keypoints. We sum the number of matched pairs between any two keypoint labels of two graphs and visualize it by heatmap. Fig. 3a shows that the baseline performs well with clean inputs while it gets fooled under attacks in Fig. 3c. All keypoints with similar appearance or structural-symmetry tend to be confused with each other, e.g. the four corners of car roof, the two sides of headlight. Fig. 3d shows that our defense mechanism helps against such attacks. Fig. 3b shows the result before/after being attacked, where the number denotes the matching probability.

adversarial attacks on keypoint locations as the *locality* attack, attacks on image pixels as the *pixel* attack, and attacks on both of them as the *combo* attack. To make our adversarial visual graph imperceptible, for pixel attack, we constrain the perturbation budget $\epsilon_{\mathbf{c}}$ as $8/255$ while for locality attack, $\epsilon_{\mathbf{z}}$ is set as 8 (the image size is 256×256). An adversarial attack example is visualized on the left of Fig. 2. It confirms the imperceptibility of our adversarial attack.

5. Appearance & Structure Aware Adversarial Training for Deep Visual Graph Matching

In this section, we first analyze the attack pattern and show that appearance-similar keypoints are more easily to be confused via statistics analysis. In Sec. 5.1, we propose a regularizer to encourage their difference. Sec. 5.2 and 5.3 show our defense mechanism using adversarial training.

5.1. Appearance Aware Regularizer

Motivation. As shown in Fig. 3b, keypoints of an object from the real-world images often contain similar appearance features, such as the four wheels of a car, on which humans depend to recognize the object. Such an appearance similarity can be summarized by three cases: shape similarity, texture similarity, and structure symmetry. Note that some similar keypoints could satisfy two or all cases, e.g. the left and right headlight of a car. We observe that under adversarial attacks, those appearance-similar keypoints are more likely to be mismatched: in Fig. 3b, the original 100% matching accuracy between two cars drops to 0% after being attacked and the attacker fools our GM solver by implicitly disturbing pairs of appearance-similar keypoints, e.g. the mismatched left and right side of the side-view mirror. Fig. 3c further validates our assumption: we attack all image pairs of "car" class and find that keypoints that are appearance-similar are intended to be mismatched with each other, which motivates us to utilize adversarial attack to discover the similarity relationship among keypoints.

Objective Design. In this paper, we propose a novel appearance aware regularizer (AAR) to explicitly enlarge the similarity among those appearance-similar keypoints in the probabilistic space of model outputs, i.e., based on the doubly-stochastic matrix $\mathbf{S} \in [0,1]^{n\times n}$ in Eq. 2. We define P=$(p_1, p_2, \dots, p_m)$, $|P|$=m as the whole set of appearance-similar groups for each graph, in which each p_i contains points which share similar appearance information.

After being attacked, we penalize those mismatched keypoints further away from the others in the same group p_i. We define an appearance aware matrix $\mathbf{R} \in \mathbb{R}^{n\times n}$ which indicates the disparity among similar keypoints:

$$\mathbf{R}_{i,j} = \begin{cases} 1.0 & \text{if } \mathbf{X}^{gt}_{i,j} = 1 \text{ and } \mathbf{X}_{i,j} \neq 1 \\ -1.0 & \text{if } \mathbf{X}^{gt}_{i,j} = 0 \text{ and } i, map(j) \in p_k, p_k \in P \\ 0.0 & otherwise \end{cases} \quad (6)$$

where $map(\cdot) : j \in [n] \mapsto i \in [n]$ projects the keypoint index in graph G^2 back to the matched index in graph G^1 based on $\mathbf{X}^{gt}$ such that the margin between i and j would get penalized in the probabilistic space if i and $map(j)$ are appearance-similar in G^1. Note that Eq. 6 focuses on those mismatched keypoints. For correctly matched keypoints of G^1 after being attacked, the corresponding row of $\mathbf{R}$ is set as 0. with no explicit penalty. Let $\mathbf{R} = (\mathbf{r}_1, \mathbf{r}_2, \dots, \mathbf{r}_n)^\top$ and each r_i means the matching probabilistic distribution of the keypoint $\mathbf{z}^1_i$ of G^1 over all keypoints $\mathbf{z}^2$ of G^2:

$$\mathbf{r}_i = \mathbf{0}, \text{ if } \mathbf{X}_i = \mathbf{X}^{gt}_i \quad (7)$$

where $\mathbf{X}_i$ and $\mathbf{X}^{gt}_i$ denote the ith row data of the two matrices. Finally, our appearance aware regularizer (AAR) is:

$$\text{AAR} = -\mathbf{R} \odot \mathbf{S} = -\sum_{i=1}^{n}\sum_{j=1}^{n} \mathbf{R}_{i,j}\mathbf{S}_{i,j} \quad (8)$$

where $\odot$ means the element-wise matrix multiplication.

Implementation. First, based on our observation in Fig. 3, we utilize the adversarial attack to discover the appearance similarity among keypoints. Fig. 4 shows a working pipeline of our proposed AAR. After getting the attacked permutation matrix, we utilize the ground-truth matrix to map the matched keypoint index in G^2 back to that in G^1. For example, we have $a \rightarrow 1$, $b \rightarrow 2$, and $c \rightarrow 0$, then

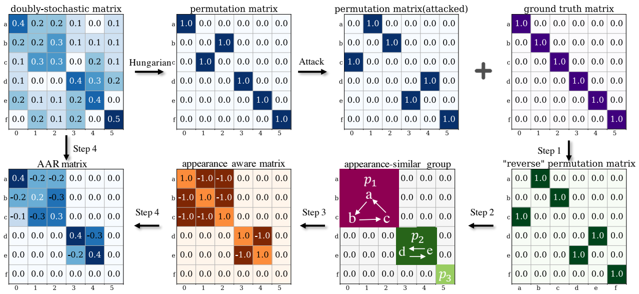


Figure 4. Pipeline of our Appearance Aware Regularizer starting from a doubly-stochastic matrix. A discrete permutation matrix is obtained via Hungarian algorithm [26]. AAR first takes the attacked permutation matrix and the ground-truth to build the "reverse" permutation matrix which reveals the matching relationship in a single graph. We next perform a depth-first search to discover the appearance-similar groups of a graph. Then we build the appearance aware matrix based on the ground-truth matrix (recall our supervised setting). Finally, we utilize that matrix to mask the doubly-stochastic matrix to obtain the AAR matrix (see Alg. 2).

after performing mapping, we have $a \rightarrow b$, $b \rightarrow c$, and $c \rightarrow a$ such that we obtain an appearance-similar group $p_1 = (a, b, c)$. Likewise, we discover other appearance-similar groups p_2 and p_3. Note that $p_3 = (f)$, which means f gets correctly matched during adversarial attacks. After we discover such groups, based on Eq. 7 and Eq. 8, we can build an AAR matrix combined with the original doubly-stochastic matrix $\mathbf{S}$ to explicitly describe the similarity margin in the probabilistic space of $\mathbf{S}$. Based on our analysis above, we define a "reverse" permutation matrix $\mathbf{X}^{rev} \in \mathbb{R}^{n\times n}$, whose element indicates the index of keypoint with which each mismatched keypoint gets matched:

$$\mathbf{X}^{rev}_{i,map(j)} = \mathbf{X}^{gt}_{i,j} \tag{9}$$

where the definition of $map(\cdot)$ follows Eq. 6. $\mathbf{X}^{rev}_{i,j} = 1$ means the mismatched keypoint i in G^1 actually gets matched with the keypoint j in G^1. Given $\mathbf{X}^{rev}$, we can perform depth first search (DFS) to discover those appearance-similar groups. The pseudo code is given in Alg. 2.

5.2. Defense Objective Design

Our defense mechanism is built on AT [30]. Based on Eq. 5, we can generate adversarial visual graphs and train our deep GM solver on those adversarial examples. Finally, we propose a new defense algorithm **A**ppearance and **S**tructure **A**ware **R**obust **G**raph **M**atching (ASAR-GM) combined with our regularization term, namely Appearance Aware Regularizer (AAR) to explicitly enlarging disparity among appearance-similar keypoints.

$$\min_{\theta} L(\mathbf{c}', \mathbf{z}', G', \mathbf{X}^{gt}; \theta) + \beta \mathrm{AAR}(\mathbf{c}', \mathbf{z}', \mathbf{X}^{gt}; \theta) \tag{10a}$$

$$s.t.\ \mathbf{c}', \mathbf{z}', G' = \arg\max_{\mathbf{c}', \mathbf{z}'} \max_{G'} L(\mathbf{c}', \mathbf{z}', G', \mathbf{X}^{gt}; \theta) \tag{10b}$$

Algorithm 2 Appearance Aware Regularizer (AAR).

Input: A pair of images $\mathbf{c} = (\mathbf{c}^1, \mathbf{c}^2)$, its keypoint sets $\mathbf{z} = (\mathbf{z}^1, \mathbf{z}^2)$, and its two graphs (G^1, G^2); NGM solver $\mathcal{M}$ and model parameters θ; perturbation budget $(\epsilon_{\mathbf{c}}, \epsilon_{\mathbf{z}})$; doubly-stochastic matrix $\mathbf{S}$, predicted permutation $\mathbf{X}$, ground-truth permutation matrix $\mathbf{X}^{gt}$.
Output: AAR **matrix.**
Obtain adversarial $\mathbf{c}'$ and $\mathbf{z}'$ via **VS-Attack** on $\mathbf{c}, \mathbf{z}$ by Alg. 1.
Attacked permutation $\mathbf{X}' \leftarrow \mathcal{M}(\mathbf{c}', \mathbf{z}'; \theta)$.
* Working pipeline of building AAR shown in Fig. 4:
1. Build "reverse" permutation $\mathbf{X}^{rev}$ in Eq. 9 by $\mathbf{X}'$ and $\mathbf{X}^{gt}$.
2. Find appearance-similar groups $P = (p_1, p_2, \ldots, p_m)$ by depth-first search on $\mathbf{X}^{rev}$.
3. Build appearance aware matrix R in Eq. 6 and Eq. 7;
4. Build AAR matrix by masking $\mathbf{S}$ with R in Eq. 8;

where the regularization term AAR follows the definition of Eq. 8, and β is the tunable scaling parameter that balances the two parts of the final loss.

5.3. Implementation

We choose the state-of-the-art GM network NGMv2 as our defense baseline. In line with NGMv2, for Eq. 10a, we adopt the binary cross entropy (BCE) as the loss:

$$L(\mathbf{c}, \mathbf{z}, G, \mathbf{X}^{gt}; \theta) = -\sum_{i=1}^{n}\sum_{j=1}^{n} \mathbf{X}^{gt}_{i,j} \log \mathbf{S}_{i,j} + (1 - \mathbf{X}^{gt}_{i,j}) \log(1 - \mathbf{S}_{i,j})$$

Since we craft our adversarial data as inputs via Eq. 10b, we also calculate AAR based on the attacked soft permutation matrix $\mathbf{S}'$. Moreover, a *burn-in period* is introduced to obtain a better trade-off between clean accuracy and robustness. We generate weaker adversarial examples in the initial period of the training process because strong adversarial examples may hurt the generalization ability of models [53] when our solver is not properly learned. We choose β as 1.5, *burn-in period* as 5 across all variants of ASAR-GM.

6. Experimental Evaluation

6.1. Evaluation Settings

Dataset. We evaluate keypoint matching on Pascal VOC dataset [14] with Berkeley annotations [4] and test the generalization ability of our method on Willow ObjectClass dataset [9]. We follow the protocol of [44] and filter out poorly annotated images and get $7,020$ training samples and $1,682$ testing samples. Experiments run on Intel(R) Xeon(R) E5-2678 v3 CPUs (2.50GHz) and 8 GTX 2080 Ti GPUs. The model is implemented by PyTorch.

Graph Matching Baselines. We validate the effectiveness of our adversarial attack over representative deep GM models: PCA-GM [45], BBGM [34], CIE-H [48], and NGMv2 [44]. For reproducibility, we apply the same training configuration of NGMv2 for defense and use the check-

Table 2. White-box robust accuracy (%) on Pascal VOC of (non-)robust models under various attacks. Adversarial examples are generated using the default loss designed for every model. "Overall" denotes mean accuracy across all data columns for each one. BBGM seems robust to current white-box attack pipeline, but it is probably due to its unique way of approximating gradients, and we show it is non-robust to black-box attack in Table 3. ASAR-GM (config 1) also boosts the accuracy of NGMv2 on clean examples.

Models	Attackers / Defenders	Clean	pixel ($\epsilon_{pix} = 8/255$)		locality ($\epsilon_{loc} = 8$)		combo ($\epsilon_{pix} = 8/255$; $\epsilon_{loc} = 8$)			Overall
			FGSM	PGD-10	FGSM	PGD-10	FGSM	PGD-10	PGD-50	
PCA-GM [42]	Baseline	64.78	25.67	10.96	41.03	30.34	23.71	9.41	7.99	26.74
CIE-H [48]	Baseline	68.92	21.80	10.24	44.62	33.04	18.89	8.82	8.16	26.81
BBGM [34]	Baseline	78.99	73.06	68.50	75.31	71.51	69.96	64.66	64.25	70.78
NGMv2 [44]	Baseline	80.40	36.97	24.59	64.78	55.54	33.51	22.41	21.46	42.46
	Pixel AT_{FGSM}	70.96	70.96	70.96	61.24	53.64	60.72	54.18	54.09	62.10
	Pixel AT_{PGD-5}	73.46	73.29	**73.20**	61.89	56.30	61.80	56.77	55.35	64.02
	Locality AT_{FGSM}	80.75	42.93	18.32	75.28	67.54	41.70	17.78	16.16	45.07
	Locality AT_{PGD-5}	80.19	38.17	13.04	73.91	70.61	36.87	12.57	15.69	42.63
	$\text{ASAR-GM}_{\text{(config 1)}}$	**81.15**	72.42	66.69	74.15	70.02	66.83	56.22	53.79	67.66
	$\text{ASAR-GM}_{\text{(config 2)}}$	79.74	**73.31**	67.42	**76.15**	**74.30**	70.14	62.39	62.11	70.70
	$\text{ASAR-GM}_{\text{(config 3)}}$	72.56	71.30	70.81	71.93	71.64	**70.70**	**69.64**	**69.60**	**71.02**

Table 3. Black-box robust accuracy (%) on Pascal VOC of (non-)robust models under various attacks. All adversarial examples are generated based on pretrained NGMv2 baseline using binary cross entropy (BCE) loss. Same "Overall" as in Table 2.

Models	Attackers / Defenders	Clean	pixel ($\epsilon_{pix} = 8/255$)		locality ($\epsilon_{loc} = 8$)		combo ($\epsilon_{pix} = 8/255$; $\epsilon_{loc} = 8$)		Overall
			FGSM	PGD-10	FGSM	PGD-10	FGSM	PGD-10	
PCA-GM [42]	Baseline	64.78	48.70	48.68	57.11	54.97	44.15	44.59	51.85
CIE-H [48]	Baseline	68.92	45.50	40.32	59.57	57.38	41.37	36.94	50.00
BBGM [34]	Baseline	78.99	52.13	47.64	71.23	68.35	47.77	44.27	58.63
NGMv2 [44]	Pixel AT_{PGD-5}	73.46	71.01	71.01	72.49	71.73	72.25	72.68	72.09
	$\text{ASAR-GM}_{\text{(config 1)}}$	**81.15**	**81.09**	**81.13**	**79.36**	78.85	79.21	**80.07**	**80.12**
	$\text{ASAR-GM}_{\text{(config 2)}}$	79.74	77.18	77.20	79.26	**79.10**	**79.49**	79.49	78.78
	$\text{ASAR-GM}_{\text{(config 3)}}$	72.56	70.41	70.41	73.26	73.21	73.25	73.28	72.34

points of other GM models collected by ThinkMatch[2].

Attack Models. We evaluate robustness of models with three types of adversarial attacks, pixel, our locality and combo attack, based on the attack scale introduced in Sec. 4.3. For each type of attack, we perform (weak) FGSM and (strong) PGD-10 attack respectively. We select PGD-50 combo attack as the possible strongest attack to benchmark the empirical lower bound of robustness. The perturbation budget is set as ϵ_{pix}=8/255 for pixel attack, ϵ_{loc}=8 for locality attack, and the same ϵ_{pix} and ϵ_{loc} for combo attack.

Defense Models. Similar to our attack models, we use adversarial training (AT) with different types of adversarial examples as our defense baseline: pixel AT with pixel attack and locality AT with locality attack. All defense baselines are also trained against adversarial data with different attack strengths from (weak) FGSM to (strong) PGD-5 attack.

6.2. Experimental Results

White-box attack results. Table 2 shows robustness of deep GM baselines and variants of NGMv2 models under white-box attacks. On the attacker side, aligned with our analysis in Sec. 4.1, our PGD-50 combo attack becomes the strongest attack among all attack baselines and consistently degrades the matching performance across all baseline models by a notable margin. For example, accuracy of NGMv2 baseline drops from 80.4% to 21.46% under this attack. On the defender side, our ASAR-GM exhibits superior robustness against all adversarial attacks compared to defense baselines. Note that we implement three versions of ASAR-GM and the only difference among them is the attack strength of adversarial data as inputs to ASAR-GM after the *burn-in period* ends. Specifically, (weaker) single-step pixel attack is applied for config 1, and (stronger) single-step combo attack is used for config 2 while (much stronger) two-step combo attack is used for config 3. ASAR-GM with config 1 achieves a better clean accuracy, 81.15% even than baseline, 80.4% while config 2 achieves better robustness with a little degradation of accuracy and config 3 further boosts robustness at a higher cost of accuracy, which agrees with the commonsense that the effectiveness of defense depends on the strength of the attack used for training [30, 36]. These results show that ASAR-GM can bring a better generalization ability on both accuracy and robustness while standard AT often suffers from the trade-off between accuracy and robustness [41,52].

Black-box attack results. We choose NGMv2 baseline model as the surrogate model and transfer adversarial examples crafted on NGMv2 to every target model. Experimental results on Table 3 demonstrate remarkable robustness of ASAR-GM against transfer-based attacks. Note that different from other baselines, robustness of BBGM drops by a notable margin when being attacked under black-box attack compared to white-box attack: accuracy under PGD-10 combo attack drops from 64.66% to 44.27%. The reason might be the gradient estimation of BBGM is a linear approximation of a piece-wise linear solver, which can be inaccurate and mislead the attacker in white-box setting.

[2] https://github.com/Thinklab-SJTU/ThinkMatch

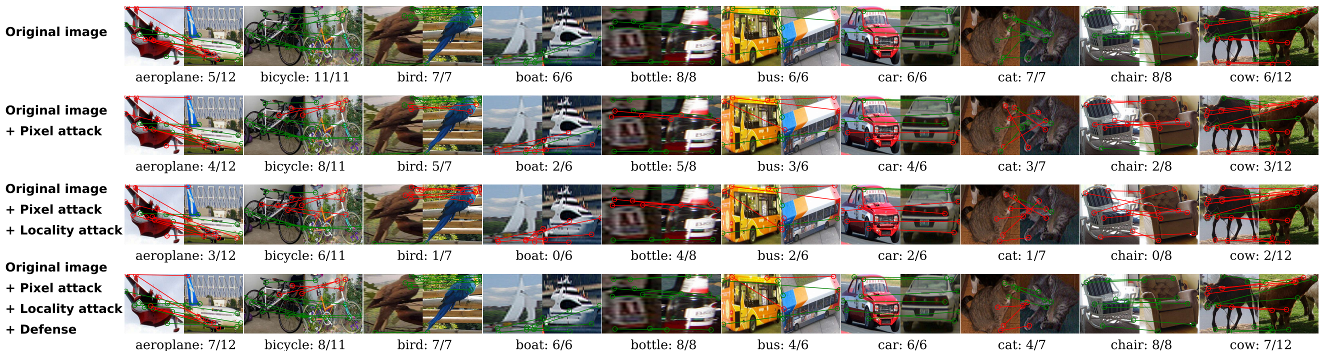


Figure 5. Visualization of the matching result of the baseline NGMv2 and our robust model under our adversarial attack. Our model exhibits superior robustness on the Pascal VOC dataset. One image pair is randomly sampled and visualized for each of the 10 classes.

Table 4. White-box robust accuracy (%) on Pascal VOC of NGMv2 as baseline for ablation study. Same attack setting in Table 2.

Defenders \ Attackers	Defense Objective	Locality Attack Type	Clean	pixel ($\epsilon_{pix} = 8/255$)		locality ($\epsilon_{loc} = 8$)		combo ($\epsilon_{pix} = 8/255$; $\epsilon_{loc} = 8$)		Overall
				FGSM	PGD-10	FGSM	PGD-10	FGSM	PGD-10	
Baseline	BCE	none	80.40	36.97	26.00	64.78	57.24	33.51	23.93	46.12
Baseline	BCE	random both	80.29	40.22	31.06	66.10	56.91	37.02	27.12	48.38
Pixel AT$_{FGSM}$	BCE	both	70.96	70.96	70.96	61.24	53.64	60.72	54.18	63.24
Pixel AT$_{FGSM}$	BCE+AAR	both	72.82	72.82	**71.92**	64.48	62.86	64.48	61.17	67.48
Locality+Pixel AT	BCE	location	79.63	73.25	68.65	72.75	69.27	67.63	58.52	69.99
Locality+Pixel AT	BCE	structure	81.67	70.69	63.05	67.93	59.05	60.90	46.38	64.24
Locality+Pixel AT	BCE	both	**81.82**	72.56	65.64	72.19	66.53	65.25	53.09	68.15
ASAR-GM$_{\text{(config 1)}}$	BCE+AAR	both	81.15	72.42	66.69	74.15	70.02	66.83	56.22	**71.92**
ASAR-GM$_{\text{(config 2)}}$	BCE+AAR	both	79.74	**73.31**	67.42	**76.15**	**74.3**	**70.14**	**62.29**	71.22

Table 5. Clean accuracy (%) on Willow ObjectClass of variants of NGMv2 models for generalization study.

method	car	duck	face	motor	bottle	mean
NGMv2 (trained on Willow ObjectCLS)	97.60	94.50	100	100	99.00	98.20
NGMv2 (trained on Pascal VOC)	80.57	75.00	99.67	66.28	94.57	83.22
ASAR-GM (trained on Pascal VOC)	89.20	81.22	99.84	82.92	98.68	90.37

Therefore, we ascribe the false sense of security of BBGM to obfuscated gradients [2].

Generalization study. Table 5 shows that ASAR-GM generalizes better from "seen" Pascal VOC to "unseen" Willow ObjectClass than standard training: improving the clean accuracy from 83.22% to 90.37%, which further corroborates that ASAR-GM learns better features of keypoints.

Ablation study. Table 4 validates the necessity of every defense component of ASAR-GM. For *locality* attack, since it directly affects graph construction, we further devise three types: "location", "structure", "both". For "location", we choose to only perturb keypoint locations while preserving the original graph structure. For "structure", we reconstruct the graph structure based on the perturbed keypoint locations while the keypoint location itself remains unchanged during inference. For "both", both keypoint locations and the following graph construction get perturbed and then join the matching pipeline together. We first implement a random version of the "both" *locality* attack and experimental results shows no benefit of such randomness on clean accuracy with little improvement of robustness. Secondly, we compare standard AT$_{FGSM}$ with(out) our regularization term, namely AAR and AAR achieves both better accuracy and robustness. Finally, for *locality* attack, compared with performance of baseline model under random attack, our perturbations on graph structure greatly improve model generalization ability, and those on both location and structure further enhance such advantage thus we choose "both" *locality* attack in our final model.

More attack baselines. To fully evaluate robustness, we conduct stronger white-box attacks with more iterations or using the target label, and another black-box attack, MI-FGSM [12]. See Appendix A for details.

Applicability of locality attack and AAR. We apply our locality attack and AAR to another baseline, PCA-GM and verify the applicability. See Appendix B for details.

Consideration of adaptive attack. By the adaptive attack criterion [39], we generate adversarial attacks via maximizing the original loss with our AAR loss together. ASAR-GM achieves 69.1% compared to PGD-50 combo attack 69.6%, signifying the robustness of our method.

7. Conclusion

In this paper, we have taken the initiative to study the vulnerability of deep (visual) GM models and design an effective adversarial attack, aiming at perturbing keypoint localities and its hidden graphs together. We further propose our defense strategy whereby an appearance-aware regularizer is developed to explicitly enlarge the disparity among the similar keypoints. Experiments on real dataset demonstrate the effectiveness of our attack and defense algorithm. Moreover, our locality attack can serve as a data augmentation and improve model generalization ability on clean test data, bringing a new SOTA on matching accuracy.

References

[1] Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein. Square attack: a query-efficient black-box adversarial attack via random search. In *Eur. Conf. Comput. Vis.*, pages 484–501, 2020. 3

[2] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *Int. Conf. Mach. Learn.*, pages 274–283, 2018. 8

[3] Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 387–402, 2013. 1

[4] Lubomir Bourdev and Jitendra Malik. Poselets: Body part detectors trained using 3d human pose annotations. In *Int. Conf. Comput. Vis.*, pages 1365–1372. IEEE, 2009. 6

[5] Jinghui Cai, Boyang Wang, Xiangfeng Wang, and Bo Jin. Accelerate black-box attack with white-box prior knowledge. In *International Conference on Intelligent Science and Big Data Engineering*, pages 394–405. Springer, 2019. 4

[6] Lichang Chen, Guosheng Lin, Shijie Wang, and Qingyao Wu. Graph edit distance reward: Learning to edit scene graph. *Eur. Conf. Comput. Vis.*, 2020. 1

[7] Yizheng Chen, Yacin Nadji, Athanasios Kountouras, Fabian Monrose, Roberto Perdisci, Manos Antonakakis, and Nikolaos Vasiloglou. Practical attacks against graph-based clustering. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1125–1142, 2017. 1

[8] Shuyu Cheng, Yinpeng Dong, Tianyu Pang, Hang Su, and Jun Zhu. Improving black-box adversarial attacks with a transfer-based prior. In *Neural Info. Process. Systems*, 2019. 4

[9] Minsu Cho, Karteek Alahari, and Jean Ponce. Learning graphs to match. In *Int. Conf. Comput. Vis.*, pages 25–32, 2013. 6

[10] Hanjun Dai, Hui Li, Tian Tian, Xin Huang, Lin Wang, Jun Zhu, and Le Song. Adversarial attack on graph structured data. In *Int. Conf. Mach. Learn.*, pages 1115–1124, 2018. 1, 2

[11] Boris Delaunay et al. Sur la sphere vide. *Izv. Akad. Nauk SSSR, Otdelenie Matematicheskii i Estestvennyka Nauk*, 7(793-800):1–2, 1934. 1, 4

[12] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Comput. Vis. Pattern Recog.*, pages 9185–9193, 2018. 8

[13] Negin Entezari, Saba A Al-Sayouri, Amirali Darvishzadeh, and Evangelos E Papalexakis. All you need is low (rank) defending against adversarial attacks on graphs. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, 2020. 1

[14] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *Int. J. Comput. Vis.*, 2010. 6

[15] Matthias Fey, Jan Eric Lenssen, Frank Weichert, and Heinrich Müller. SplineCNN: Fast geometric deep learning with continuous b-spline kernels. In *Comput. Vis. Pattern Recog.*, pages 869–877, 2018. 3

[16] Matthias Fey, Jan E Lenssen, Christopher Morris, Jonathan Masci, and Nils M Kriege. Deep graph matching consensus. In *Int. Conf. Learn. Rep.*, 2020. 1

[17] Quankai Gao, Fudong Wang, Nan Xue, Jin-Gang Yu, and Gui-Song Xia. Deep graph matching under quadratic constraint. In *Comput. Vis. Pattern Recog.*, pages 5069–5078, 2021. 1, 2

[18] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv:1412.6572*, 2014. 1, 2, 3

[19] Chuan Guo, Jacob Gardner, Yurong You, Andrew Gordon Wilson, and Kilian Weinberger. Simple black-box adversarial attacks. In *Int. Conf. Mach. Learn.*, pages 2484–2493, 2019. 3

[20] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. In *Int. Conf. Mach. Learn.*, pages 2137–2146, 2018. 3

[21] Jinyuan Jia, Binghui Wang, Xiaoyu Cao, and Neil Zhenqiang Gong. Certified robustness of community detection against adversarial structural perturbation via randomized smoothing. In *Proceedings of The Web Conference*, pages 2718–2724, 2020. 2

[22] Shuai Jia, Yibing Song, Chao Ma, and Xiaokang Yang. Iou attack: Towards temporally coherent black-box adversarial attack for visual object tracking. In *Comput. Vis. Pattern Recog.*, pages 6709–6718, 2021. 2, 3

[23] Bo Jiang, Pengfei Sun, Jin Tang, and Bin Luo. Glmnet: Graph learning-matching networks for feature matching. *PR*, 2021. 1, 2

[24] Bo Jiang, Pengfei Sun, Ziyan Zhang, Jin Tang, and Bin Luo. Gamnet: Robust feature matching via graph adversarial-matching network. In *ACM MM*, pages 5419–5426, 2021. 1

[25] Hongwei Jin, Zhan Shi, Venkata Jaya Shankar Ashish Peruri, and Xinhua Zhang. Certified robustness of graph convolution networks for graph classification under topological attacks. In *Neural Info. Process. Systems*, 2020. 2

[26] Harold W Kuhn. The hungarian method for the assignment problem. In *Export. Naval Research Logistics Quarterly*, pages 83–97, 1955. 6

[27] Jia Li, Honglei Zhang, Zhichao Han, Yu Rong, Hong Cheng, and Junzhou Huang. Adversarial attack on community detection by hiding individuals. In *Proceedings of The Web Conference*, pages 917–927, 2020. 1

[28] Siyuan Liang, Xingxing Wei, Siyuan Yao, and Xiaochun Cao. Efficient adversarial attacks for visual object tracking. *Eur. Conf. Comput. Vis.*, 2020. 3

[29] Eliane Maria Loiola, Nair Maria Maia de Abreu, Paulo Oswaldo Boaventura-Netto, Peter Hahn, and Tania Querido. A survey for the quadratic assignment problem. *Eur. J. Operational Research*, 176(2):657–690, 2007. 3

[30] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017. 2, 3, 4, 6, 7

[31] Alex Nowak, Soledad Villar, Afonso S Bandeira, and Joan Bruna. Revised note on learning quadratic assignment with graph neural networks. In *Data Science Workshop*, 2018. 2

[32] Talal Rahwan, Marcin Waniek, Tomasz P Michalak, Yevgeniy Vorobeychik, and Kai Zhou. Attacking similarity-based link prediction in social networks. In *Proceedings of the 18th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, 2019. 1

[33] Jiaxiang Ren, Zijie Zhang, Jiayin Jin, Xin Zhao, Sixing Wu, Yang Zhou, Yelong Shen, Tianshi Che, Ruoming Jin, and Dejing Dou. Integrated defense for resilient graph matching. In *Int. Conf. Mach. Learn.*, pages 8982–8997, 2021. 1, 2, 3

[34] Michal Rolínek, Paul Swoboda, Dominik Zietlow, Anselm Paulus, Vít Musil, and Georg Martius. Deep graph matching via blackbox differentiation of combinatorial solvers. In *Eur. Conf. Comput. Vis.*, pages 407–424. Springer, 2020. 1, 2, 3, 6, 7

[35] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *Trans. Neural Netw.*, 20(1):61–80, 2008. 3

[36] Gaurang Sriramanan, Sravanti Addepalli, Arya Baburaj, and R Venkatesh Babu. Guided adversarial attack for evaluating and enhancing adversarial defenses. *arXiv preprint arXiv:2011.14969*, 2020. 7

[37] Yiwei Sun, Suhang Wang, Xianfeng Tang, Tsung-Yu Hsieh, and Vasant Honavar. Adversarial attacks on graph neural networks via node injections: A hierarchical reinforcement learning approach. In *Proceedings of The Web Conference*, pages 673–683, 2020. 1, 2

[38] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013. 1

[39] Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. In *Neural Info. Process. Systems*, 2020. 8

[40] Binghui Wang and Neil Zhenqiang Gong. Attacking graph-based classification via manipulating the graph structure. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 2023–2040, 2019. 1

[41] Haotao Wang, Tianlong Chen, Shupeng Gui, TingKuei Hu, Ji Liu, and Zhangyang Wang. Once-for-all adversarial training: In-situ tradeoff between robustness and accuracy for free. In *Neural Info. Process. Systems*, pages 7449–7461, 2020. 7

[42] Runzhong Wang, Junchi Yan, and Xiaokang Yang. Learning combinatorial embedding networks for deep graph matching. In *Int. Conf. Comput. Vis.*, pages 3056–3065, 2019. 1, 2, 3, 7

[43] Runzhong Wang, Junchi Yan, and Xiaokang Yang. Combinatorial learning of robust deep graph matching: an embedding based approach. *IEEE TPAMI*, 2020. 2, 3

[44] Runzhong Wang, Junchi Yan, and Xiaokang Yang. Neural graph matching network: Learning lawler's quadratic assignment problem with extension to hypergraph and multiple-graph matching. *IEEE TPAMI*, 2021. 1, 2, 3, 6, 7

[45] Tao Wang, He Liu, Yidong Li, Yi Jin, Xiaohui Hou, and Haibin Ling. Learning combinatorial solver for graph matching. In *Comput. Vis. Pattern Recog.*, pages 7568–7577, 2020. 6

[46] Yu Xiong, Qingqiu Huang, Lingfeng Guo, Hang Zhou, Bolei Zhou, and Dahua Lin. A graph-based framework to bridge movies and synopses. In *Int. Conf. Comput. Vis.*, 2019. 1

[47] Kaidi Xu, Hongge Chen, Sijia Liu, Pin-Yu Chen, Tsui-Wei Weng, Mingyi Hong, and Xue Lin. Topology attack and defense for graph neural networks: An optimization perspective. *arXiv preprint arXiv:1906.04214*, 2019. 2

[48] Tianshu Yu, Runzhong Wang, Junchi Yan, and Baoxin Li. Learning deep graph matching with channel-independent embedding and hungarian attention. In *Int. Conf. Learn. Rep.*, 2020. 1, 2, 3, 6, 7

[49] Tianshu Yu, Runzhong Wang, Junchi Yan, and Baoxin Li. Deep latent graph matching. In *Int. Conf. Mach. Learn.*, pages 12187–12197. PMLR, 2021. 1

[50] Yu-Feng Yu, Guoxia Xu, Min Jiang, Hu Zhu, Dao-Qing Dai, and Hong Yan. Joint transformation learning via the l2, 1-norm metric for robust graph matching. *IEEE transactions on cybernetics*, 2019. 3

[51] Andrei Zanfir and Cristian Sminchisescu. Deep learning of graph matching. In *Comput. Vis. Pattern Recog.*, pages 2684–2693, 2018. 1, 2, 3

[52] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In *Int. Conf. Mach. Learn.*, pages 7472–7482, 2019. 7

[53] Jingfeng Zhang, Xilie Xu, Bo Han, Gang Niu, Lizhen Cui, Masashi Sugiyama, and Mohan Kankanhalli. Attacks which do not kill training make adversarial learning stronger. In *Int. Conf. Mach. Learn.*, pages 11278–11287, 2020. 6

[54] Xiang Zhang and Marinka Zitnik. Gnnguard: Defending graph neural networks against adversarial attacks. *arXiv preprint arXiv:2006.08149*, 2020. 2

[55] Zhen Zhang and Wee Sun Lee. Deep graphical feature learning for the feature matching problem. In *Int. Conf. Comput. Vis.*, pages 5087–5096, 2019. 2, 3

[56] Zijie Zhang, Zeru Zhang, Yang Zhou, Yelong Shen, Ruoming Jin, and Dejing Dou. Adversarial attacks on deep graph matching. In *Neural Info. Process. Systems*, 2020. 1, 2, 3

[57] Kaixuan Zhao, Shikui Tu, and Lei Xu. Ia-gm: A deep bidirectional learning method for graph matching. In *AAAI Conf. Artificial Intell.*, volume 35, pages 3474–3482, 2021. 1, 2

[58] Dingyuan Zhu, Ziwei Zhang, Peng Cui, and Wenwu Zhu. Robust graph convolutional networks against adversarial attacks. In *SIGKDD*, pages 1399–1407, 2019. 2

[59] Daniel Zügner and Stephan Günnemann. Adversarial attacks on graph neural networks via meta learning. *arXiv preprint arXiv:1902.08412*, 2019. 1