# PROSPECT: Learn MLPs on Graphs Robust against Adversarial Structure Attacks

Bowen Deng
dengbw3@mail2.sysu.edu.cn
Sun Yat-sen University
Guangzhou, China

Jialong Chen
chenjlong7@mail2.sysu.edu.cn
Sun Yat-sen University
Guangzhou, China

Yanming Hu
huym27@mail2.sysu.edu.cn
Sun Yat-sen University
Guangzhou, China

Zhiyong Xu
xuzhy63@mail2.sysu.edu.cn
Sun Yat-sen University
Guangzhou, China

Chuan Chen*
chenchuan@mail.sysu.edu.cn
Sun Yat-sen University
Guangzhou, China

Tao Zhang*
zhangt358@mail.sysu.edu.cn
Sun Yat-sen University
Guangzhou, China

## Abstract

Current adversarial defense methods for GNNs exhibit critical limitations obstructing real-world application: **1)** inadequate adaptability to graph heterophily, **2)** absent generalizability to early GNNs like GraphSAGE used downstream, and **3)** low inference scalability unacceptable for resource-constrained scenarios. To simultaneously address these challenges, we propose the first online GNN-MLP distillation framework PROSPECT, which merges the complementary knowledge of MLP and GNN and can thus learn GNN and MLP robust against adversarial structure attacks on both homophilic and heterophilic graphs. PROSPECT integrates seamlessly into GraphSAGE and achieves inference scalability exponentially higher than conventional GNNs. To mitigate potential convergence failure caused by inductive bias conflicts between the heterogeneous MLP and GNN, we propose the Quasi-Alternating Cosine Annealing (QACA) learning rate scheduler, inspired by our convergence analysis of the involved MLP. Experiments on homophilic and heterophilic graphs demonstrate the advantages of PROSPECT over current defenses and offline GNN-MLP distillation methods in terms of adversarial robustness and clean accuracy, the inference scalability of PROSPECT orders of magnitude higher than existing defenses, and the effectiveness of QACA.

## CCS Concepts

• **Security and privacy**; • **Computing methodologies** → **Machine learning**;

## Keywords

Adversarial machine learning; graph neural network; graph knowledge distillation; graph heterophily

*Corresponding authors.


*CIKM '24, October 21–25, 2024, Boise, ID, USA*

ACM ISBN 979-8-4007-0436-9/24/10
https://doi.org/10.1145/3627673.3679857

**ACM Reference Format:**
Bowen Deng, Jialong Chen, Yanming Hu, Zhiyong Xu, Chuan Chen, and Tao Zhang. 2024. PROSPECT: Learn MLPs on Graphs Robust against Adversarial Structure Attacks. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24), October 21–25, 2024, Boise, ID, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3627673.3679857

## 1 Introduction

Graph (network) data are ubiquitous today, playing a fundamental role across disciplines including computer science [13] and social science [4]. In recent years, GNNs [17, 23, 51] have emerged as the most promising tools for graph data analysis. GNNs have been utilized in a wide array of domains, such as drug discovery [21], network routing [41], financial risk management [28], e-commerce [59], social media [34], and recommendation systems [15]. Despite their enormous success, GNNs have been shown susceptible, like other deep learning models, to malicious data perturbations known as adversarial attacks [10, 60, 61]. The key distinction between GNNs and non-graph models lies in the graph structure that connects entities. This also differentiates the study of adversarial attacks and robustness in GNNs. Perturbing the graph structure is more harmful than altering node features, as edges affect all feature dimensions and feature manipulations can be mitigated by GNN Neighborhood Aggregation [50]. Thus, we focus on adversarial robustness against structure attacks, which are classified into two types based on their occurrence stages: **1)** evasion attacks during inference/testing, and **2)** poisoning attacks during training.

As graph attacks alter properties such as homophily[1], many defenses (e.g., purification defenses) detect and counter adversarial attacks by identifying these induced changes. Entezari et al. [14] find Nettack [60] substantially impacts high-rank adjacency singular components, motivating low-rank approximation as the clean graph. Wu et al. [50] observe that attacks insert heterophily, inspiring purification based on Jaccard similarity [20] between node pairs. Similarly, GNNGuard [55] prunes the edges linking dissimilar nodes at every layer. Jin et al. [22] summarize these findings into that real-world clean graphs should admit sparsity, smoothness (i.e., homophily), and low rankness, and accordingly propose to learn such graphs while training GNNs. Leveraging graph contrastive learning [29, 46], STABLE [25] learns node features robust

[1]Heterophily/homophily means that neighboring nodes tend to have different/similar labels and features

to structure perturbations and then refines the graph according to node feature similarities. Although these methods are robust on homophilic graphs, their performance on heterophilic graphs has not been studied. In principle, these robust mechanisms designed for homophilic graphs are difficult to effectively adapt to heterophilic graphs, as demonstrated by [11] and our experiments.

Different from the above structure purification methods, some works develop heterophily-aware GNNs to handle heterophily inherent in the graph itself or caused by attacks. Zhu et al. [57] show heterophilic GNNs like $H_2$GCN [58] are more robust than homophilic GNNs such as GCN [23] and GAT [45]. Lei et al. [24] propose EvenNet to generalize across different homophily levels and thus defend attacks. However, these robust heterophilic GNNs are designed ad-hoc. Thus downstream GNNs used in recommendation [18, 53] or security [39] cannot benefit, since they commonly build on simple GNNs like GraphSAGE (SAGE) [17]. In addition to heterophilic GNNs, the purification method GARNET [12] estimates a highly homophilic structure through the top-$k$ dominant adjacency singular components. It does not rely on homophily prior but depends on the $k$-truncated singular value decomposition ($k$-TSVD), which is impractical for scaling to large graphs.

Beyond adversarial robustness, inference speed is also critical for industrial applications. GNNs often exhibit prohibitive latency due to their graph dependency, as adding layers necessitates aggregating information from increasingly distant neighbors. This is expensive as inference for one node in an $L$-layer SAGE incurs an exponential complexity of $O(D^L)$, given the average node degree $D$. To overcome this barrier, offline GNN-MLP distillation methods [43, 54] transfer knowledge from GNNs to MLPs for fast inference. However, the capabilities of offline GNN-MLPs are constrained by the GNN teacher and the unidirectional knowledge transfer. Moreover, these offline methods are vulnerable to structure poisoning attacks, as evidenced by our experiments (Section 5).

We now summarize that previous defenses are less or more challenged by the problems below.

- **Inadequate heterophily adaptability.** Most of purification defenses do not consider inherent heterophily and thus exhibit deficient adaptability to heterophilic graphs.
- **Absent generalizability.** The robustness mechanisms of heterophilic GNNs are ad hoc, lacking generalizability to the downstream models built on early GNNs.
- **High inference latency.** All current defenses inference with graph structure. So like most GNNs, the inference latency is too high to handle high-pressure situations.

In response to the aforementioned limitations, this paper introduces a novel online bidirectional GNN-MLP knowledge distillation method that integrates the knowledge of two heterogeneous models, specifically MLP and GNN. This fusion of knowledge allows for adaptability to graph heterophily, enhances adversarial robustness, generalizes well to naive SAGE, and maintains the same inference scalability as MLP. The key motivation underneath is that heterogeneous models like MLP and GNN emphasize different aspects of graph data (node features versus graph structure), thus providing complementary information. As illustrated in Table 1, MLP and SAGE have distinct sets of correct predictions on both clean and attacked graph data, resulting in a significantly larger union of correct predictions than either model alone. MLP is better than SAGE on clean heterophilic UAI, and the performance gap is widened under MetaAttack [61]. This implies that the knowledge from MLP is beneficial to GNN, but traditional GNNs and offline GNN-MLP distillation methods cannot transfer the knowledge from MLP to GNN. To bridge this gap, we propose incorporating MLP-to-GNN knowledge distillation, leading to our mutual and online GNN-MLP distillation framework PROSPECT.

**Table 1: MLP and SAGE have complementary knowledge. The test accuracy (%) on (poisoned) homophilic Cora and heterophilic UAI is reported. The datasets are chosen from Section 5.2. The MLP-SAGE row corresponds to the union of correct predictions from SAGE and MLP.**

| | Cora | Cora-Meta-15 | UAI | UAI-Meta-15 |
|---|---|---|---|---|
| MLP | 59.96 | 59.96 | 61.08 | 61.08 |
| SAGE | 83.50 | 71.38 | 55.09 | 41.24 |
| MLP-SAGE | 88.68 | 82.80 | 70.09 | 66.91 |

Transferring MLP knowledge to GNN through mutual distillation appears feasible, but in practice, we find it difficult due to the significant differences in inductive bias between heterogeneous models. Since the MLP's prediction results $\mathbf{Z}_m$ do not contain graph structural information, they usually differ significantly from the GNN's prediction results $\mathbf{Z}_g$. When $\mathbf{Z}_m$ and $\mathbf{Z}_g$, which differ considerably, are used as distillation targets for each other, the entire system is pulled in two different directions, leading to optimization difficulties. To address these issues, we analyze the convergence conditions of Prospect-MLP in Theorem 1 and propose the quasi-alternating cosine annealing (QACA) learning rate scheduler.

Our method **PROSPECT** is the first online and mutual GNN-MLP distillation framework to learn GNNs and ML**P**s **RO**bust again**S**t gra**P**h adv**E**rsarial stru**C**ture at**T**acks. We denote the engaged MLP and GNN as Prospect-MLP and Prospect-GNN, respectively, while referring to the corresponding instance that employs SAGE [17] as Prospect-SAGE. Our main contributions can be summarized as follows.

- As far as we are aware, we propose the *first* online GNN-MLP distillation framework PROSPECT, which incorporates an adversarial robustness mechanism catering to both homophilic and heterophilic graphs, enables seamless integration with SAGE, and achieves inference scalability orders of magnitude higher than conventional GNNs.
- To the best of our knowledge, we are the *first* to investigate the adversarial robustness of GNN-MLP distillation methods, revealing the vulnerability of previous offline ones to poisoning attacks. In contrast, the proposed online and mutual GNN-MLP framework PROSPECT is robust against both evasion and poisoning structure attacks.
- We discover the optimization challenges that heterogeneous GNN-MLP mutual distillation may face, and analyze the convergence of Prospect-MLP in Theorem 1. Inspired by this analysis, we design the QACA learning rate scheduler.

- Experiments on five homophilic and three heterophilic graphs not only validate the effectiveness of QACA but also demonstrate the superior adversarial robustness, clean accuracy, heterophily adaptability, and inference scalability of PROSPECT over baselines.

## 2 Related Work and Notations

### 2.1 GNN Defense Methods

Previous adversarial defenses fall into four types. **1) Adversarial training.** Perturbing the clean adjacency matrix with random flips [10], gradient projection descent [52], or Nettack [7] during training can confer some evasion attack robustness. But it may impede training efficiency, fail to withstand poisoning attacks, and risk clean accuracy vs robustness trade-offs [35]. **2) Preprocess purification.** The susceptible components, like high-rank adjacency components [14] or dissimilar connections [50], are removed before training/inference. GARNET [11], which estimates the clean graph based on top-$k$ largest singular components that are hardly affected by adversarial attacks, partially addresses the heterophily adaptation problem. But its scalability is limited by the $k$-TVSD. **3) Learning purification.** Learning clean graphs during training can be done by assigning low propagation weights for susceptible elements [56], attenuating edges connecting dissimilar nodes [55], optimizing a dense adjacency matrix towards the properties of clean homophilic graphs [22], and extracting robust node features for subsequent reconstruction [25]. **4) Heterophilic design.** Many attack algorithms, e.g., [60, 61], insert heterophily into homophilic graphs [22, 25, 50, 55, 57], to degrade homophilic GNNs. In contrast, GNNs designed for heterophilic graphs, including $H_2$GCN [58] and EvenNet [24], can more or less adapt to the altered homophily levels. Hence, they exhibit some inherent robustness against current adversarial attacks [57].

Compared to type 1 models, PROSPECT defends against both poisoning and evasion attacks without any potential accuracy-robustness trade-offs. Unlike types 2 and 3, PROSPECT inherently adapts to heterophily, requiring no additional purification costs and incurring minimal training overhead. Versus type 4, PROSPECT enables integration with simple GNNs [18, 34, 53] used downstream, rather than being ad-hoc. And unlike all four types, PROSPECT has an inference scalability as high as MLPs. Most crucially, the adversarial robustness of PROSPECT stems from the knowledge fusion without any assumptions, basically suitable for any graph and any structure attacks.

### 2.2 GNN-MLP Distillation

PROSPECT pioneers online and mutual GNN-MLP distillation frameworks, versus offline and unidirectional ones like GLNN [54] and NOSMOG [43]. GLNN transfers the GNN knowledge learned from the graph structure and node features to MLPs that rely on no graph structures, by matching the temperatured logits [19, 38]. Such design is, however, shown unable to align the input node feature to the label space fully, resist node feature noises, and capture the soft structural representational similarity among nodes. To address these problems, NOSMOG incorporates the structure embeddings, e.g., DeepWalk [37], into node features, distills the relative node similarity [44], and employs Project Gradient Descent adversarial training (PGD-AT) [31] on node features to tackle noises.

The differences between these offline GNN-MLPs and PROSPECT are as follows. **1)** GLNN and NOSMOG are vulnerable to poisoning structure and evasion node feature attacks, while PROSPECT resists both poisoning and evasion structure attacks. **2)** The performance of offline methods is constrained by the pre-trained teachers, whereas PROSPECT transcends this limit through mutual distillation. **3)** PROSPECT simultaneously trains robust GNNs and MLPs in one phase, avoiding the complex two-phase of offline distillation. **4)** PROSPECT concerns the structure adversarial robustness, which is neglected by NOSMOG but more destructive and prevalent in the graph machine learning context.

### 2.3 Notations

Given an undirected and connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ consisting of $N$ nodes $\mathcal{V} = \{1, \cdots, N\}$ and $M = |\mathcal{E}|$ edges , we denote the adjacency and degree matrices respectively by $\mathbf{A}$ and $\mathbf{D} = \text{diag}\,(\mathbf{A} \cdot \mathbf{1})$, wherein $\mathbf{1}$ is an all-one column vector with appropriate length and $\text{diag}\,(\mathbf{r})$ generates a diagonal matrix taking vector $\mathbf{r}$ as the diagonal. Since $\mathbf{A}$ summaries the structure information in $\mathcal{V}$ and $\mathcal{E}$, the graph data with node feature matrix $\mathbf{X}$ can be comprehensively described by the tuple $\mathcal{G} = (\mathbf{A}, \mathbf{X})$, where the $i$-th row of $\mathbf{X} \in \mathbb{R}^{N \times d}$ is the transpose of $d$-dimensional node feature column vector of the $i$-th node.

Real-world graphs often exhibit varying degrees of homophily, which significantly impacts the GNN (and defense) performance. There are various ways to quantify homophily [27, 36, 58], and we adopts the most widely accepted one [58] described in Definition 1, following [57].

**Definition 1.** *(homophily ratio, HR) Given a $N$-node graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and node label vector $\mathbf{y} \in \mathbb{R}^N$, the edge-based homophily ratio is defined as the fraction of edges linking same-label nodes*

$$h(\mathcal{G}, \mathbf{y}) = \frac{1}{|\mathcal{E}|} \sum_{(i,j) \in \mathcal{E}} \mathbb{1}(y_i, y_j), \tag{1}$$

*where $\mathbb{1}(y_i, y_j) = 1$ when $y_i = y_j$ and 0 otherwise.*

The pioneering GNN model GraphSAGE [17] are widely used downstream and can be formulated as (without sampling)

$$\mathbf{H}^{(l+1)} = \phi\left(\mathbf{H}^{(l)}\mathbf{W}_1^{(l)} + \mathbf{P}\mathbf{H}^{(l)}\mathbf{W}_2^{(l)}\right), \tag{2}$$

where $\phi(\cdot)$ is the activation function, $\mathbf{H}^{(l)}$ denotes the input features of the $l$-th layer, $\mathbf{P} = \mathbf{D}^{-1}\mathbf{A}$ is the propagation matrix and $\mathbf{W}^{(l)}$ is the weight matrix. An $L$-layer GNN is a function mapping the $d$-dimensional input node features $\mathbf{H}^{(0)} = \mathbf{X} \in \mathbb{R}^{N \times d}$ to the normalized logits $\mathbf{Z} = f_\theta(\mathcal{G}) = f_\theta(\mathbf{X}, \mathbf{A}) \in \mathbb{R}^{N \times C}$ over $C$ classes.

## 3 PROSPECT Framework

The knowledge fusion between GNN and MLP is performed by incorporating both GNN-to-MLP and MLP-to-GNN disiatlltion, as illustrated in Figure 1. The optimization objective of PROSPECT

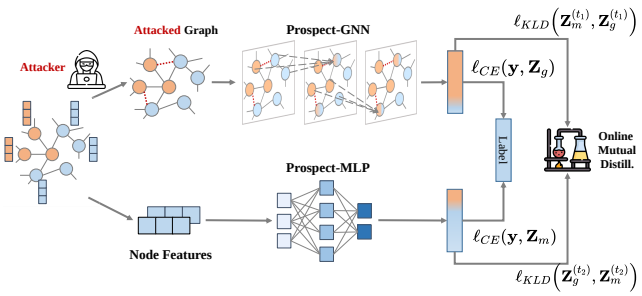



**Figure 1: PROSPECT architecture. MLP and GNN independently learn data knowledge with cross-entropy loss while simultaneously distilling knowledge to each other. After training, either Prospect-GNN or Prospect-GNN can be deployed.**

comprises GNN Eq. (3b) and MLP Eq. (3c) parts, and can be formulated as

$$\mathcal{L}_{pro} = \mathcal{L}_g + \mathcal{L}_m \tag{3a}$$

$$\mathcal{L}_g = \frac{1}{|\mathcal{V}_L|}\sum_{i\in\mathcal{V}_L} \ell_{CE}\left(y_i, [\mathbf{Z}_g]_i\right) + \frac{\alpha_1 t_1^2}{|\mathcal{V}_{obs}|}\sum_{i\in\mathcal{V}} \ell_{KLD}\left(\left[\mathbf{Z}_m^{(t_1)}\right]_i, \left[\mathbf{Z}_g^{(t_1)}\right]_i\right) \tag{3b}$$

$$\mathcal{L}_m = \frac{1}{|\mathcal{V}_L|}\sum_{i\in\mathcal{V}_L} \ell_{CE}\left(y_i, [\mathbf{Z}_m]_i\right) + \frac{\alpha_2 t_2^2}{|\mathcal{V}_{obs}|}\sum_{i\in\mathcal{V}} \ell_{KLD}\left(\left[\mathbf{Z}_g^{(t_2)}\right]_i, \left[\mathbf{Z}_m^{(t_2)}\right]_i\right), \tag{3c}$$

where $\ell_{CE}$ is the cross-entropy loss, $\ell_{KLD}$ is the Kullback-Leibler divergence (KLD), $\mathcal{V}_L$ is the training set and $\mathcal{V}_{obs}$ is the set of nodes with observable features, $\alpha_1$ and $\alpha_2$ are the weights of distillation losses, the subscripts of $\mathbf{Z}_g$ and $\mathbf{Z}_m$ denote the prediction matrices separately belonging to Prospect-GNN and Prospect-MLP, and the superscripts $t_1$ and $t_2$ of $\mathbf{Z}^{(t_1)}$ and $\mathbf{Z}^{(t_2)}$ are softmax temperatures.

## 3.1 Adversarial robustness

The trained Prospect-GNN and Prospect-MLP usually achieve comparable performance (as shown by the experiments in Section 5). Thus we prefer to deploy the later for fast inference. Since MLP does not require graph structure, *PROSPECT can be completely immune to evasion structure attacks*. Note that due to such perfect evasion robustness, the performance of Prospect-MLP under evasion attacks will be the same as when trained and tested on clean graphs. Robustness against evasion node feature attacks can be addressed through efficient adversarial feature training, which has been extensively explored in non-graph literature. Therefore, we do not focus on this aspect.

In the context of poisoning attacks, GNNs are compromised by the tainted structure during training, whereas MLPs only utilize clean node features. Consequently, MLP-to-GNN distillation can cleanse the erroneous knowledge in Prospect-GNN by leveraging the pure knowledge from the cooperative Prospect-MLP. To prevent the MLP from being poisoned by GNNs, we employ an alternating learning strategy (detailed in Section 4): primarily training the MLP first, followed by primarily training the GNN within each period.

## 3.2 Clean accuracy improvement

Offline GNN-MLP methods, like GLNN [54], improve the MLP's to match that of the GNN teacher by transferring structural knowledge through the GNn-to-MLP distillation direction. Since PROSPECT includes this distillation direction, Prospect-MLP will at least achieve the performance level of GLNN. Furthermore, recent studies on non-graph data [26] indicate that the key for online distillation to surpass offline distillation lies in the reverse distillation from the student to the teacher. Reverse distillation can reduce the knowledge gap and thus facilitates knowledge transfer from teacher to student. As PROSPECT incorporates MLP-to-GNN distillation, it is likely that Prospect-MLP will outperform the unidirectional GLNN.

For Prospect-GNN, it is known that GNNs primarily focus on structural information in graphs, sometimes resulting in insufficient exploitation of node feature information, particularly when deeper layers lead to indistinguishable embeddings [6, 32, 40]. In contrast, MLPs consistently excel in extracting node features. Thus, Prospect-MLP can help Prospect-GNN outperform the standalone GNN by imparting knowledge of discriminative feature learning.

## 3.3 Heterophily Adaptability

Previous defenses [14, 22, 25, 50, 55, 56] utilize feature similarity between node pairs to detect adversarial edges, operating under the premise that attacks decrease homophily. This cannot differentiate between clean and adversarial edges on heterophilic graphs, since the clean graph itself is already highly heterophilic. PROSPECT, on the other hand, transfers clean node knowledge from the MLP to the GNN on a node-by-node basis, thus mitigating the effects of poisoned structures without relying on pairwise comparisons. As a result, PROSPECT does not require graph homophily assumption.

Beyond adversarial robustness, the improvements in clean accuracy are also observed on heterophilic graphs. Many GNNs produce similar features for neighboring nodes, leading to the tendency of classifying adjacent nodes into the same category, which aids in classification on homophilic graphs. However, on heterophilic graphs, neighboring nodes often belong to different categories. In such cases, we need less similar representations for adjacent nodes and MLP can provide such representations to regulate the GNN.

## 3.4 Inference Scalability

After training, Prospect-MLP can be deployed to latency-sensitive industrial scenarios [9, 33]. Since MLP does not rely on the graph structure, the majority of time and space consumption arsed from neighbor fetching and aggregation is saved.

# 4 QACA Scheduler

The PROSPECT optimization objective Eq. (3a) consists of four potentially conflicting components, making the learning more complex than with MLP or GNN alone. To mitigate potential convergence issues caused by knowledge conflicts between heterogeneous MLP and GNN, we investigate the convergence conditions

of Prospect-MLP in Theorem 1 and adopt the cosine annealing (CA) [30] learning accordingly. Additionally, we use an alternating strategy, silencing each participant in turn to further stabilize the training process. Together, these two ingredients constitute our QACA learning rate scheduler.

## 4.1 QACA Design

**Theorem 1.** *Given a Prospect-MLP trained with the loss function Eq. (3c) and assume that* $\exists u > 0$,

$$\mathrm{Tr}\left\{(w_1 - w_2)^\top \left[\nabla \mathcal{L}_m(w_1) - \nabla \mathcal{L}_m(w_2)\right]\right\} \geq u\|w_1 - w_2\|_F^2, \quad (4)$$

*one global or local optimal MLP weight* $w^*$ *of the last layer can be found by gradient descent if*

$$0 \leq \left(1 + \eta^2\beta^2 - 2\eta u\right) < 1 \quad (5a)$$

$$\beta = \frac{1}{|\mathcal{V}_L|}\sigma(\mathbf{H}^\top \mathbf{S}^\top \mathbf{S})\sigma(\mathbf{H}) + \frac{\alpha t_2}{N}\sigma^2(\mathbf{H}), \quad (5b)$$

*where* $\eta$ *is the learning rate,* $\mathbf{H}$ *is the input feature matrix of last MLP layer,* $\sigma(\cdot)$ *is the matrix spectral norm, and* $\mathbf{S} \in \{0,1\}^{|\mathcal{V}_L| \times N}$ *is the row selection matrix to extract the training node rows of the matrix* $\mathbf{X} \in \mathbb{R}^{N \times d}$ *into* $\mathbf{SX} \in \mathbb{R}^{|\mathcal{V}_L| \times d}$ .

Theorem 1, proved in Section 4.2, establishes the relationship between the learning rate $\eta$, the input to the final MLP layer, the distillation weight $\alpha$, and the distillation temperature $t_2$ when the model converges. The convergence condition Eq. (5a) is crucial in inspiring the design of QACA. Besides, the value range of $\mu$ in the assumption Eq. (4) vary throughout the training, with changes diminishing as the model approaches convergence.

*4.1.1 Cosine Annealing (CA).* Although Theorem 1 specifically addresses the final MLP layer, the primary proof methodology can be extended to derive similar quadratic inequalities for all layers. Nonetheless, this simplified theorem provides sufficient motivation for our approach. Defining $g(\eta) = \eta^2\beta^2 - 2\eta u$, the convergence inequality (5a) becomes $0 > g(\eta) \geq -1$. The roots of $g(\eta)$ are 0 and $2u/\beta^2$, and the points $\left(\frac{u-\sqrt{u^2-\beta^2}}{\beta^2}, -1\right)$ and $\left(\frac{u+\sqrt{u^2-\beta^2}}{\beta^2}, -1\right)$ lie on $g(\eta)$ if $u > \beta$. Figure 2 depicts two scenarios of $g(\eta)$, highlighting a gap between the feasible regions in the left scenario. For a given learning rate $\eta_0$, the potential outcomes in the left scenario include:

- $\eta_0 > \frac{2u}{\beta^2}$: it does not meet the convergent condition
- $\frac{u-\sqrt{u^2-\beta^2}}{\beta^2} < \eta_0 < \frac{u+\sqrt{u^2-\beta^2}}{\beta^2}$: it does not satisfies the convergent condition
- $\frac{u+\sqrt{u^2-\beta^2}}{\beta^2} \leq \eta_0 \leq \frac{2u}{\beta^2}$ or $0 < \eta_0 \leq \frac{u-\sqrt{u^2-\beta^2}}{\beta^2}$ : as the training progresses, $\eta_0$ may fall outside these two regions because $u$ and $\beta$ typically fluctuate during the training process.

For the right scenario depicted in Figure 2, similar challenges arise, albeit with different feasible regions. To mitigate these issues, one simple approach is to select a very small $\eta_0$ near the origin, ensuring that $\eta_0$ always remains within the feasible region, even as $u$ and $\beta$ fluctuate. However, this can result in unacceptably slow training.

To overcome this problem, we propose employing an annealing strategy during each scheduling period. By starting with a suitably large initial learning rate $\eta_0$ and a cold lower bound $\eta_{\min}$ near the origin, we enable rapid training in the initial epochs. The learning rate then adapts to remain within the feasible regions for a greater number of epochs compared to a fixed $\eta_0$, as it can traverse or stay within these regions even as they change with $u$ and $\beta$ during training. Ultimately, $\eta$ decays to $(0, \eta_{\min}]$, where $\eta_{\min} \ll \left(u - \sqrt{u^2 - \beta^2}\right)/\beta^2$, an interval that is relatively insensitive to variations $u$ and $\beta$. Given the well-documented benefits of fast convergence and improved accuracy facilitated by cosine annealing with warm restarts [30], our annealing component is built on this scheduling method.

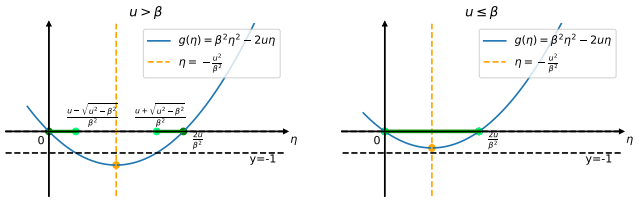



**Figure 2: Plot of $g(\eta) = \eta^2\beta^2 - 2\eta u$. The orange lines are the symmetry axes.The green segments on the horizontal axis are the feasible regions of the convergence inequality in (5a).**

*4.1.2 Quasi-alternating (QA) learning.* The model heterogeneity between GNN and MLP in PROSPECT may cause knowledge conflicts, which can be further exacerbated by poisoned structures. When both learning rates are high, the rapid and intense exchange of knowledge can confuse the participants in PROSPECT. Conversely, if only one learning rate is high, the unique knowledge from the side with the lower learning rate will remain relatively stable and thus more accessible. Additionally, as indicated by Theorem 1, alternating the learning rates can stabilize the training dynamics of Prospect-MLP. Specifically, when Prospect-GNN remains inactive and maintains stable knowledge, $u$ in Eq. (4) is less likely to oscillate because the gradients from GNN-to-MLP distillation are more stable with consistent GNN knowledge. Furthermore, drawing an analogy to the alternating iterative turbo decoding [2], we hypothesize that an alternating knowledge exchange mechanism can help PROSPECT mitigate errors caused by poisoning attacks.

*4.1.3 QACA learning rate scheduler.* The above analysis results in our QACA scheduler, which can be formulated as

$$\eta_T = \begin{cases} \eta_{\min} + \frac{(\eta_{\max} - \eta_{\min})}{2}\left(1 + \cos\left(\frac{2T_{\text{cur}}\,\pi}{T_0}\right)\right) & T_{\text{cur}} < \frac{T_0}{2} \\ \eta_{\min} & T_{\text{cur}} \geq \frac{T_0}{2} \end{cases}, \quad (6)$$

where $\eta_{\min}$ and $\eta_{\max}$ determine the range of learning rates, $T_{\text{cur}} = (T+B) \mod T_0$ accounts for how many epochs since the last restart, $B$ is the offset before starting scheduling, and $T_0$ epochs constitute a minimal schedule period. Within one minimal period, QACA performs annealing in the first $T_0/2$ epochs and silences learning in the later half. In PROSPECT, we set the offsets $B = T_0/2$ for GNN and $B = 0$ for MLP to train them alternatingly. Besides, a small $\eta_{\min}$ retains some model activity, making learning"quasi"-alternating.

## 4.2 Proof of Theorem 1

*4.2.1 Auxiliaries for Theorem 1 Proof .* The tools required to prove Theorem 1 are listed here, with their proofs provided in the supplementary materials.

**Definition 2.** *(Lipschitz constant) A function $f : \mathcal{X} \to \mathcal{Y}$ is K-Lipschitz (continuous) w.r.t. a norm $\|\cdot\|$ if there is a constant K such that*

$$\forall x_1, x_2 \in \mathcal{X}, \|f(x_1) - f(x_2)\| \leq K\|x_1 - x_2\|. \tag{7}$$

*The smallest K admits the inequality is one of the Lipschitz constants of $f$ and denoted as $\|f\|_{Lip}$.*

**Theorem 2** (Rademacher [16], Theorem 3.1.6; [47], Theorem 1)**.** *If $f : \mathbb{R}^n \to \mathbb{R}^m$ is a locally Lipschitz continuous function[2] , then $f$ is differentiable almost everywhere. Moreover, if $f$ is Lipschitz continuous, then*

$$\|f\|_{Lip} = \sup_{x \in \mathbb{R}^n} \|\nabla f(x)\|_2, \tag{8}$$

*where $\|\mathbf{M}\|_2 = \sup_{\|x\| \leq 1} \|\mathbf{M}x\|_2$ is the operator norm of matrix $\mathbf{M} \in \mathbb{R}^{m \times n}$.*

**Theorem 3** (Banach fixed-point theorem)**.** *Let $(\mathcal{X}, \mathcal{D})$ be a non-empty complete metric space with a contraction mapping $f : \mathcal{X} \to \mathcal{X}$ such that*

$$\mathcal{D}\left(f(x_1), f(x_2)\right) \leq q\mathcal{D}\left(x_1, x_2\right), \exists q \in [0, 1), \tag{9}$$

*then there is a unique fixed-point $f(x^*) = x^*$ that can be found by generating a sequence $\{x_n \mid x_{n+1} = f(x_n)\}_{n \in \mathbb{N}}$ with an initial point $x^{(0)} \in \mathcal{X}$. $\mathcal{D}$ is usually a vector or matrix norm.*

**Proposition 1.** *Given two functions $f : \mathcal{X} \to \mathcal{U}$ and $g : \mathcal{U} \to \mathcal{Y}$ whose Lipschitz constants are respectively $\|g\|_{Lip}$ and $\|f\|_{Lip}$, the Lipschitz constant of the composite function $g \circ f : \mathcal{X} \to \mathcal{Y}$ satisfies the inequality*

$$\|g \circ f\|_{Lip} \leq \|g\|_{Lip}\|f\|_{Lip}. \tag{10}$$

**Proposition 2.** *The Lipschitz constant w.r.t. the Frobenius norm of the linear operator $\mathbf{Y}_{m \times n} = \mathbf{A}_{m \times k}\mathbf{X}_{k \times n}$ is the spectral norm of $\mathbf{A}$, and that of the row-wise t-softmax function $\mathbf{Y}_{m \times n} = \text{softmax}_t(\mathbf{X}_{m \times n})$ is $1/t$.*

*4.2.2 Main Proof of Theorem 1 .*

Proof. For the symbol simplicity, we replace $\mathbf{W}$ with $w$ here. The local or global optimal MLP weight $w^*$ should satisfy stationary point condition $\nabla_w L_m(w^*) = 0$, meaning that $w^* = w^* - \eta\nabla_w \mathcal{L}_m(w^*)$. We can thus construct a function $G(w) = w - \eta\nabla_W \mathcal{L}_m(w)$ where $\eta$ is the step size of gradient decent and $w^*$ is the fixed-point of $G(w)$. Let the metric $\mathcal{D}$ be Frobenius matrix norm. If follows that for any two weights $w_1, w_2 \in \mathcal{W}$

$$\begin{aligned}
&\|G(w_1) - G(w_2)\|_F^2 && \text{(11a)}\\
=&\|w_1 - w_2 - \eta\left(\nabla\mathcal{L}_m(w_1) - \nabla\mathcal{L}_m(w_2)\right)\|_F^2 && \text{(11b)}\\
=&\|w_1 - w_2\|_F^2 + \eta^2\|\nabla\mathcal{L}_m(w_1) - \nabla\mathcal{L}_m(w_2)\|_F^2 \\
&- 2\,\text{Tr}\left\{\eta(w_1 - w_2)^\top\left[\nabla\mathcal{L}_m(w_1) - \nabla\mathcal{L}_m(w_2)\right]\right\}. && \text{(11c)}
\end{aligned}$$

To apply Banach fixed-point theorem, we need construct the inequality between $\|G(w_1) - G(w_2)\|_F^2$ and $\|w_1 - w_2\|_F^2$, so the second and third terms in (11c) should be tackled. Since the assumption in Theorem 1 copes with the third term, we move on to the second term now.

The gradient of Prospect-MLP loss function $\mathcal{L}_m$ (Eq. (3c)) w.r.t. the weights of all layers can be obtained by backpropagation. For simplicity, we denote by $f_t(\cdot)$ the last $t$-softmax layer and only consider the weight of the last MLP layer. In spite of this, our proof can be extended to an arbitrary MLP layer since the main tool, i.e., Proposition 1, is extendable[3]. However, such extension would lead to cumbersome formulas without providing additional insights. To proceed, we expand the second term to

$$\begin{aligned}
&\nabla\mathcal{L}_m(w_1) - \nabla\mathcal{L}_m(w_2)\\
=&\frac{\mathbf{H}^\top}{N_{tr}}\mathbf{S}^\top\mathbf{S}\left(f(\mathbf{H}w_1) - \mathbf{Y}\right) + \frac{\alpha t_2}{N}\mathbf{H}^\top\left(f_{t_2}(\mathbf{H}w_1) - \mathbf{Z}_g^{t_2}\right)\\
&- \frac{\mathbf{H}^\top}{N_{tr}}\mathbf{S}^\top\mathbf{S}\left(f(\mathbf{H}w_2) - \mathbf{Y}\right) - \frac{\alpha t_2}{N}\mathbf{H}^\top\left(f_{t_2}(\mathbf{H}w_2) - \mathbf{Z}_g^{t_2}\right) && \text{(12a)}\\
=&\frac{\mathbf{H}^\top}{N_{tr}}\mathbf{S}^\top\mathbf{S}\left[f(\mathbf{H}w_1) - f(\mathbf{H}w_2)\right] + \frac{\alpha t_2}{N}\mathbf{H}^\top\left[f_{t_2}(\mathbf{H}w_1) - f_{t_2}(\mathbf{H}w_2)\right], && \text{(12b)}
\end{aligned}$$

where $\mathbf{H}$ is the input feature matrix of last MLP layer and $N_{tr} = |\mathcal{V}_L|$ is the size of training set.

We construct two functions

$$g_1(w) = \frac{\mathbf{H}^\top}{N_{tr}}\mathbf{S}^\top\mathbf{S}f(\mathbf{H}w) \tag{13}$$

$$g_2(w) = \frac{\alpha t_2}{N}\mathbf{H}^\top f_{t_2}(\mathbf{H}w), \tag{14}$$

and then Eq. (12b) turns out to be

$$\nabla\mathcal{L}_m(w_1) - \nabla\mathcal{L}_m(w_2) = g_1(w_1) - g_1(w_2) + g_2(w_1) - g_2(w_2). \tag{15}$$

It follows that

$$\begin{aligned}
&\|\nabla\mathcal{L}_m(w_1) - \nabla\mathcal{L}_m(w_2)\|_F^2 && \text{(16a)}\\
=&\|g_1(w_1) - g_1(w_2) + g_2(w_1) - g_2(w_2)\|_F^2 && \text{(16b)}\\
=&\|g_1(w_1) - g_1(w_2)\|_F^2 + \|g_2(w_1) - g_2(w_2)\|_F^2 \\
&+ 2\|g_1(w_1) - g_1(w_2)\|_F\|g_2(w_1) - g_2(w_2)\|_F && \text{(16c)}\\
\leq&\|g_1(w)\|_{Lip}^2\|w_1 - w_2\|_F^2 + \|g_2(w)\|_{Lip}^2\|w_1 - w_2\|_F^2 \\
&+ 2\|g_1(w)\|_{Lip}\|g_2(w)\|_{Lip}\|w_1 - w_2\|_F^2 && \text{(16d)}
\end{aligned}$$

According to Propositions 2 and 1, we have

$$\|g_1(w)\|_{Lip} = \frac{1}{N_{tr}}\sigma(\mathbf{H}^\top\mathbf{S}^\top\mathbf{S})\sigma(\mathbf{H}) \tag{17}$$

$$\|g_2(w)\|_{Lip} = \frac{\alpha t_2}{N}\sigma^2(\mathbf{H}). \tag{18}$$

Then the upper bound (19) becomes

$$U_1 = \left[\frac{1}{N_{tr}}\sigma(\mathbf{H}^\top\mathbf{S}^\top\mathbf{S})\sigma(\mathbf{H}) + \frac{\alpha t_2}{N}\sigma^2(\mathbf{H})\right]^2\|w_1 - w_2\|_F^2. \tag{19}$$

[2]The functions whose restriction to some neighborhood around any point is Lipschitz are *locally* Lipschitz.

[3]The extended proof also employs that the common activation functions (e.g., ReLU, LeakyReLU, Sigmoid, Tanh, and Sigmoid) are both 1-Lipschitz.

Substituting Eq. (19) and the assumption Eq. (4) into Eq. (11c) leads to

$$\|G(w_1) - G(w_2)\|_F^2 \tag{20a}$$

$$\leq \|w_1 - w_2\|_F^2 + \eta^2 \left[\frac{1}{N_{tr}}\sigma(\mathbf{H}^\top \mathbf{S}^\top \mathbf{S})\sigma(\mathbf{H}) + \frac{\alpha t_2}{N}\sigma^2(\mathbf{H})\right]^2 \|w_1 - w_2\|_F^2 \tag{20b}$$

$$= \left\{1 + \eta^2 \left[\frac{1}{N_{tr}}\sigma(\mathbf{H}^\top \mathbf{S}^\top \mathbf{S})\sigma(\mathbf{H}) + \frac{\alpha t_2}{N}\sigma^2(\mathbf{H})\right]^2 - 2\eta u\right\} \|w_1 - w_2\|_F^2 \tag{20c}$$

$$= \left(1 + \eta^2\beta^2 - 2\eta u\right) \|w_1 - w_2\|_F^2. \tag{20d}$$

□

# 5 Experiments

In this section, extensive experiments on a variety of datasets are conducted to answer the following research questions (RQs). **RQ1**: How robust is the proposed PROSPECT? **RQ2**: Does PROSPECT compromise clean accuracy? **RQ3**: Does PROSPECT possess high heterophily adaptability? **RQ4**: Can QACA facilitate the training process of PROSPECT? **RQ5**: How scalable is the inference of PROSPECT?

## 5.1 Experimental Settings

**Table 2: The data statistics of the used graphs.**

| Dataset | #Nodes | #Edges | #Features | #Classes | HR |
|---|---|---|---|---|---|
| Texas | 183 | 279 | 1703 | 5 | 0.061 |
| Polblogs | 1222 | 16714 | 1490 | 2 | 0.906 |
| Citeseer | 2110 | 3668 | 3703 | 6 | 0.736 |
| Chameleon | 2277 | 31371 | 2325 | 5 | 0.230 |
| Cora | 2485 | 5069 | 1433 | 7 | 0.804 |
| CoraML | 2810 | 7981 | 2879 | 7 | 0.784 |
| ACM | 3025 | 13128 | 1870 | 3 | 0.821 |
| UAI | 3067 | 28311 | 4973 | 19 | 0.364 |

*5.1.1 Datasets.* We consider public graph datasets: Cora, Citeseer, UAI [42], ACM [48], Polblogs [1], Chameleon, Texas [36] and CoraML [3].The statistics of the largest connected components of these graphs are summarized in Table 2. Following [61], the largest connected component (LCC) of each graph is taken and split with 10% nodes for training, 10% validation, and 80% testing. We repeat such 1:1:8 data splitting with 5 random seeds on each graph, and the results averaged over these 5 distinct splits are reported as the eventual performance on that graph. Attacked graphs are produced to evaluate the model robustness and the attackers usually have only limited attack budgets to modify the graph data. An attack budget of 5% (15%) means the attacker can flip $0.05|\mathcal{E}|$ ($0.15|\mathcal{E}|$) entries of adjacency matrix.

*5.1.2 Baselines.* The baselines are diverse. Method only using node features: MLP. Early simple GNNs: GCN [23], SAGE [17], and SGC [49]. Purification-based adversarial defenses: SVD [14], Jaccard [50], RGCN [56], Guard [55], ProGNN [22], STABLE [25], and GARNET [12]. Heterophilic GNNs: GPRGNN [8] and EvenNet [24]. Offline GNN-MLP distillation: GLNN [54]. The hidden dimension size of MLPw4 is 4 times that of MLP. The hidden dimension size of GLNNw4 is 4 times that of GLNN.

*5.1.3 More Details.* Please refer to the supplementary materials for details on the software and hardware environment, the code, and the hyperparameter search range. Additionally, the supplementary materials include the full robustness results for all 8 datasets under 4 different attack budgets and the inference scalability results on two additional datasets not covered in the main text. The code is available at https://github.com/bwdeng20/PROSPECT

## 5.2 Adversarial Robustness (RQ1 & RQ3)

MetaAttack (with a GCN surrogate) [61], is an effective structure attack algorithm and serves as a key benchmark for evaluating robustness. We use MetaAttack to generate contaminated graphs under different attack budgets and five random splits. For instance, a Citeseer dataset attacked with a 15% budget is denoted as Citeseer-Meta-15. Subsequently, we use these contaminated datasets to perform **transfer poisoning** attacks on GNNs and defenses. For each attack budget, we report the mean and standard deviation of results across five random splits. Results for attack budgets of 5% and 15% are presented in Table 3, while those for 10% and 20% can be found in the supplementary materials.

On heterophilic UAI and Texas, consistent with the seed experiments in Section 1, the accuracy of MLP significantly surpasses that of GNNs, particularly GCN, especially under attack. Additionally, as discussed in Section 1, purification defenses designed based on graph homophily priors (ranging from RGCN to STABLE in Table 3) perform poorly on heterophilic graphs, sometimes even worse than unprotected SGC and SAGE. Among heterophilic GNNs, EvenNet performs better than GPR, indicating that the balance theory [5] inductive bias is more robust to MetaAttack than GPR's higher-order neighborhood aggregation.

Surprisingly, even when the teacher SAGE is poisoned, GLNN cultivates a more robust student via naive offline distillation [19]. Some errors seems to be filtered out in distillation. This has several implications: 1) Existing graph attacks appear ineffective against distillation methods; 2) Appropriately designed offline knowledge distillation could be a simple trick to enhance GNN robustness.

Across all combinations of four datasets and two attack budgets shown in Table 3, Prospect-SAGE and Prospect-MLP take the top and second positions. Whether on homophilic graphs (Polblogs and Citeseer) or heterophilic graphs (UAI and Texas), the robustness of Prospect-SAGE and Prospect-MLP significantly surpasses that of standalone SAGE and MLP, demonstrating that the robustness mechanisms of PROSPECT are suited to heterophilic graphs.

Regarding evasion robustness, the performance of Prospect-MLP will be equivalent to clean performance (Table 4). Considering that Prospect-MLP already, on clean data, matches or surpasses other models affected by evasion structure attacks, its advantage will significantly increase when actual evasion attacks occur due to its perfect evasion robustness. That is, Prospect-MLP is almost definitely more robust than others regarding evasion structure attacks. Therefore, we skip the real evasion robustness evaluation.

**Table 3: The robustness results (%) on four datasets attacked by MetaAttack. The top two performing models are highlighted in bold, with the best further underlined.**

| | Polblogs (HR=0.906) | | Citeseer (HR=0.736) | | UAI (HR=0.364) | | Texas (HR=0.061) | |
|---|---|---|---|---|---|---|---|---|
| | 5% | 15% | 5% | 15% | 5% | 15% | 5% | 15% |
| MLP | 52.21±0.61 | | 66.01±1.37 | | 61.74±2.11 | | 65.71±4.42 | |
| MLPw4 | 51.72±0.87 | | 66.43±1.73 | | 62.71±1.91 | | 68.98±3.32 | |
| GCN | 77.18±1.76 | 67.53±0.99 | 72.03±1.23 | 64.74±2.70 | 56.72±4.68 | 54.22±3.17 | 49.25±5.43 | 49.39±2.29 |
| SGC | 77.71±1.79 | 66.95±1.36 | 71.94±1.31 | 64.51±2.44 | 58.78±3.34 | 56.52±2.64 | 53.88±2.23 | 55.24±2.12 |
| SAGE | 90.39±0.66 | 77.34±3.74 | 72.68±1.25 | 70.40±1.05 | 60.02±3.21 | 60.18±2.65 | 62.99±3.39 | 64.35±2.99 |
| RGCN | 75.42±1.29 | 66.18±0.64 | 71.71±2.04 | 64.02±1.90 | 49.89±2.85 | 48.40±2.74 | 52.93±1.89 | 49.52±8.10 |
| SVD | 92.43±0.70 | 73.44±1.77 | 69.82±0.86 | 65.15±2.01 | 48.65±1.14 | 44.87±1.18 | 49.66±4.02 | 48.57±5.66 |
| Jaccard | 50.88±1.69 | 50.88±1.69 | 72.18±1.81 | 66.96±2.71 | 54.08±4.18 | 50.64±2.69 | 49.25±5.43 | 49.39±2.29 |
| Guard | 51.58±0.57 | 51.58±0.57 | 69.79±1.24 | 67.35±0.62 | 20.28±10.99 | 20.36±8.27 | 48.03±12.96 | 47.76±11.40 |
| ProGNN | 85.97±5.16 | 72.78±3.43 | 71.60±1.84 | 65.12±2.38 | 49.22±5.22 | 38.43±11.55 | 47.89±10.06 | 45.31±14.47 |
| STABLE | 92.80±2.38 | 88.55±0.38 | 74.33±1.08 | 73.32±1.14 | 51.78±2.08 | 47.63±2.26 | 52.27±2.82 | 50.52±3.24 |
| EvenNet | 87.04±1.45 | 68.06±1.50 | 74.08±1.02 | 70.95±1.71 | 67.8±2.029 | 66.91±2.18 | 62.45±2.70 | 63.27±2.85 |
| GPR | 69.45±1.08 | 56.13±2.01 | 73.40±1.04 | 69.82±1.89 | 35.38±9.24 | 34.75±11.11 | 54.15±2.75 | 51.02±7.37 |
| GPR-GARNET | 72.91±0.84 | 59.57±1.74 | 74.05±0.80 | 74.49±1.50 | 32.39±5.32 | 28.17±2.75 | 54.97±6.39 | 57.55±3.95 |
| GLNN | 91.62±1.35 | 77.46±3.73 | 74.25±1.20 | 71.92±1.38 | 62.46±2.91 | 62.02±2.00 | 66.40±2.57 | 66.53±5.45 |
| GLNNw4 | 91.19±1.41 | 77.12±3.58 | 74.01±1.32 | 71.94±1.38 | 62.62±2.39 | 62.75±1.99 | 67.21±2.70 | 66.40±4.92 |
| Prospect-SAGE | **93.95±1.34** | **92.27±2.26** | **75.01±0.75** | **74.81±0.41** | **69.86±0.58** | **69.52±0.46** | **68.84±5.65** | **71.02±2.30** |
| Prospect-MLP | **93.99±0.76** | **93.95±0.34** | **75.31±1.18** | **74.79±0.64** | **68.31±0.59** | **69.10±0.45** | **72.11±2.06** | **73.20±1.53** |

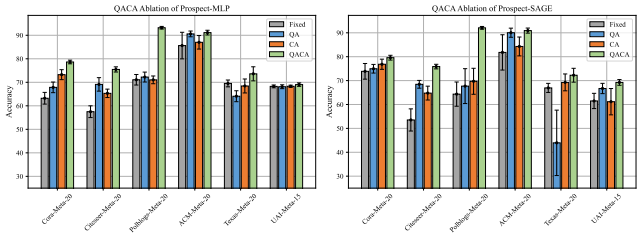
**Table 4: Accuracy (%) comparison on clean graphs. The mean and std over five splits are reported. The top three performing models are highlighted in bold, with the best further underlined.**

| | homophilic (HR>0.5) | | | | | heterophilic (HR<0.5) | | |
|---|---|---|---|---|---|---|---|---|
| | Cora | Citeseer | Polblogs | ACM | CoraML | Texas | Chameleon | UAI |
| MLP | 65.69±1.43 | 66.02±1.37 | 52.21±0.61 | 87.44±0.28 | 71.31±0.65 | 65.71±4.42 | 42.65±0.86 | 61.74±2.11 |
| MLPw4 | 67.02±1.12 | 66.43±1.73 | 51.72±0.87 | 87.39±0.69 | 71.60±0.94 | **68.98±3.32** | 41.08±2.05 | 62.71±1.91 |
| GCN | 84.20±0.92 | 73.40±2.01 | 94.79±1.20 | 89.65±0.75 | 85.94±0.68 | 51.29±6.46 | **56.69±2.68** | 63.51±1.29 |
| SGC | 82.58±0.75 | 73.57±1.49 | 94.68±0.90 | 90.03±0.79 | 84.66±0.64 | 53.06±1.86 | 50.91±2.30 | 62.28±2.32 |
| SAGE | 83.56±0.86 | 74.53±1.01 | 94.40±0.77 | 90.31±0.90 | 84.51±1.09 | 64.76±1.76 | 51.66±2.21 | 60.56±3.83 |
| RGCN | 83.85±0.63 | 72.94±1.68 | 94.87±0.81 | 89.40±2.08 | 86.19±0.57 | 51.57±2.17 | **55.74±1.46** | 54.80±1.68 |
| SVD | 77.72±0.37 | 69.65±1.53 | 93.58±0.85 | 86.41±1.49 | 81.09±0.61 | 51.02±2.85 | 47.87±2.25 | 50.58±1.82 |
| Jaccard | 82.95±0.68 | 73.50±1.72 | 50.88±1.69 | 89.65±0.75 | 84.81±0.47 | 51.29±6.46 | 45.32±1.27 | 61.09±1.05 |
| Guard | 78.33±1.15 | 70.14±2.30 | 51.58±0.57 | 89.23±1.14 | 77.06±1.07 | 48.44±8.61 | 40.89±2.27 | 32.84±2.31 |
| ProGNN | 83.84±0.77 | 73.72±0.99 | 94.83±0.51 | 90.17±0.76 | 85.64±0.73 | 51.84±3.31 | 53.63±1.39 | 57.65±1.01 |
| STABLE | 83.09±0.58 | 74.44±0.56 | 94.68±0.45 | 85.40±0.83 | 83.62±0.46 | 50.27±4.70 | 46.66±1.57 | 56.47±0.48 |
| EvenNet | **84.89±0.35** | 74.46±0.80 | **95.24±0.55** | 90.54±0.57 | **86.48±0.31** | 67.21±1.22 | 51.73±1.22 | **70.07±1.16** |
| GPR | 84.43±0.83 | 74.88±1.23 | 94.68±0.43 | 92.13±0.70 | **86.38±0.66** | 56.33±8.76 | 51.30±0.87 | 34.99±1.93 |
| GPR-GARNET | 83.61±0.59 | 74.87±0.58 | 93.03±0.77 | **92.35±0.48** | 85.98±0.90 | 56.87±2.91 | 50.31±0.91 | 33.53±7.85 |
| GLNN | 83.17±0.68 | 75.14±0.84 | 94.15±0.63 | 91.90±0.45 | 84.87±0.86 | 67.48±3.38 | 48.36±2.19 | 62.54±3.34 |
| GLNNw4 | 83.23±0.79 | **75.60±0.52** | 94.58±0.82 | 91.89±0.51 | 84.99±1.00 | 68.44±4.60 | 49.36±2.07 | 62.94±2.79 |
| Prospect-SAGE | **84.94±0.51** | **75.20±0.70** | **95.22±0.24** | **93.15±0.86** | 85.93±0.91 | **72.79±3.22** | **55.88±1.12** | **69.90±0.92** |
| Prospect-MLP | **84.50±0.58** | **75.81±0.68** | **95.32±0.41** | **93.22±0.71** | **86.54±0.75** | **73.06±1.64** | 53.43±1.45 | **68.97±0.66** |

## 5.3 Clean Accuracy (RQ2 & RQ3)

In adversarial attack research on non-graph data, there is typically a tradeoff between a defense model's adversarial robustness and clean accuracy [35]. Although adversarial training is not used, GNN defenses also show a slight tendency towards this tradeoff. In the clean performance Table 4, purification methods (from RGCN to STABLE) generally have slightly lower clean accuracy compared to their protected GCN counterparts. This is primarily because edges that benefit classification are likely to be mistakenly removed as adversarial edges during the purification process.

Our proposed PROSPECT achieves robustness through knowledge fusion, a mechanism that does not compromise the information of clean graphs. So PROSPECT does not sacrifice any clean




(a) Prospect-MLP (b) Prospect-SAGE

**Figure 3: Ablation study of QACA learning rate scheduling. Fixed indicates no learning rate change, QA enables quasi-alternating learning, CA utilizes the cosine annealing with warm restart [30], and QACA combines QA and CA.**

performance. On the contrary, due to the applicability of the knowledge fusion mechanism to clean graphs, both Prospect-SAGE and Prospect-MLP show significant performance improvements over their respective SAGE and MLP baselines, with increases of up to about 9 percentage points (on UAI and Texas). Additionally, as shown in Table 4, PROSPECT is effective on both homophilic and heterophilic graphs, confirming its adaptability to heterophily.

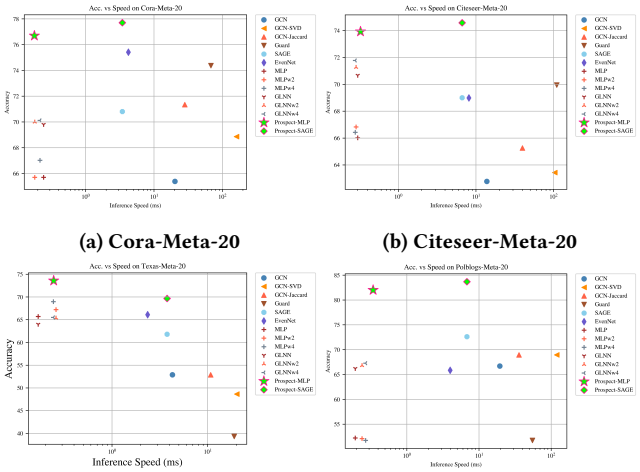
## 5.4 QACA Effectiveness (RQ4)

In Section 1, we discussed the potential optimization difficulties of GNN-MLP mutual distillation. To demonstrate this, we conducted experiments with the following setups: 1) training PROSPECT with a fixed learning rate. Given that QACA scheduler comprises two components—cosine annealing (CA) and quasi-alternating (QA), we designed two ablation optimization baselines: 2) using only the cosine annealing learning rate strategy without alternating updates for GNN and MLP; 3) employing a fixed learning rate but periodically alternating updates for GNN and MLP. We compare the performance of these three optimization methods with QACA across multiple attacked datasets. Figure 3 shows that, on most datasets, using either QA or CA alone results in better performance than a fixed learning rate. And, combining QA and CA (i.e., QACA) consistently outperforms all other optimization strategies across all datasets.

## 5.5 High Inference Scalability (RQ5)

The real-world production is with a semi-inductive scenario [54] typically lying between transductive and inductive settings. In this scenario, some test nodes are visible during training (transductive test nodes), while others are only visible during testing (inductive test nodes). Under this setup, we employ MetaAttack to target the whole graph, resulting in a mixed attack combining elements of both poisoning and evasion, as the test graph is also perturbed. We then randomly split the test set $\mathcal{V}_{test} = \mathcal{V}_{test}^{trans} \cup \mathcal{V}_{test}^{ind}$, and measure the average inference speed on 10 inductive test nodes, along with accuracy on $\mathcal{V}_{test}$.

Figure 4 demonstrates that despite not utilizing graph structure during inference, Prospect-MLP achieves semi-inductive robustness comparable to Prospect-SAGE, significantly outperforming other models. More importantly, Prospect-MLP's inference speed




(a) Cora-Meta-20 (b) Citeseer-Meta-20

(c) Texas-Meta-20 (d) Polblogs-Meta-20

**Figure 4: Acc. (%) vs. inference speed (ms) in the semi-inductive setting like [54]. That is 20% test nodes are excluded as inductive ones during training and validation while 80% test nodes are transductive ones observable across all stages. The x-axis is logarithmically scaled.**

is markedly faster than all methods except MLP. For instance, on Cora-Meta-20, Prospect-MLP achieves robustness far superior to GCN-SVD while delivering 500 times faster inference speed.

## 6 Conclusions

To address key limitations of existing GNN defenses: **1)** inadequate adaptability to heterophily; **2)** absent generalizability to early GNNs such as SAGE; **3)** low inference scalability, this study provides simple yet efficient PROSPECT. PROSPECT pioneers online and mutual GNN-MLP distillation that merges the complementary knowledge between GNN and MLP. It can inference as efficiently as MLPs and seamlessly fit into early GNNs like SAGE. We analyzes the potential GNN-MLP knowledge conflicts from the convergence perspective in Theorem 1, which inspires our QACA scheduler. Experiments on five homophilic and three heterophilic graph datasets demonstrate the effectiveness of QACA scheduler, the high inference scalability of PROSPECT, and the superior adversarial robustness and clean accuracy of PROSPECT over previous defenses and offline GNN-MLP distillation methods.

Compared to the methods proposed, the more significant contributions of this study lie in the insights and inspirations it offers. We uncover the complementary nature of MLP and GNN knowledge and demonstrate that this knowledge can be fused through mutual distillation guided by the QACA scheduler. Our findings prompt a reevaluation of the potential of MLPs in graph learning and extend the application of GNN-MLP distillation beyond mere inference acceleration.

## Acknowledgments

The research is supported by the National Natural Science Foundation of China (62176269).

## References


[1] Lada A. Adamic and Natalie Glance. 2005. The Political Blogosphere and the 2004 U.S. Election: Divided They Blog. In *Proceedings of the 3rd International Workshop on Link Discovery (LinkKDD '05)*. Association for Computing Machinery, New York, NY, USA, 36–43. https://doi.org/10.1145/1134271.1134277

[2] C. Berrou, A. Glavieux, and P. Thitimajshima. 1993. Near Shannon Limit Error-Correcting Coding and Decoding: Turbo-codes. 1. In *Proceedings of ICC '93 - IEEE International Conference on Communications*, Vol. 2. 1064–1070 vol.2. https://doi.org/10.1109/ICC.1993.397441

[3] Aleksandar Bojchevski and Stephan Günnemann. 2018. Deep Gaussian Embedding of Graphs: Unsupervised Inductive Learning via Ranking. In *International Conference on Learning Representations*.

[4] Stephen P. Borgatti, Ajay Mehra, Daniel J. Brass, and Giuseppe Labianca. 2009. Network Analysis in the Social Sciences. *Science* 323, 5916 (Feb. 2009), 892–895. https://doi.org/10.1126/science.1165821

[5] Dorwin Cartwright and Frank Harary. 1956. Structural Balance: A Generalization of Heider's Theory. *Psychological Review* 63, 5 (1956), 277–293. https://doi.org/10.1037/h0046049

[6] Deli Chen, Yankai Lin, Wei Li, Peng Li, Jie Zhou, and Xu Sun. 2020. Measuring and Relieving the Over-Smoothing Problem for Graph Neural Networks from the Topological View. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 3438–3445. https://doi.org/10.1609/aaai.v34i04.5747

[7] Jinyin Chen, Yangyang Wu, Xiang Lin, and Qi Xuan. 2019. Can Adversarial Network Attack Be Defended? *CoRR* abs/1903.05994 (2019).

[8] Eli Chien, Jianhao Peng, Pan Li, and Olgica Milenkovic. 2022. Adaptive Universal Generalized PageRank Graph Neural Network. In *International Conference on Learning Representations*.

[9] Avery Ching, Sergey Edunov, Maja Kabiljo, Dionysios Logothetis, and Sambavi Muthukrishnan. 2015. One Trillion Edges: Graph Processing at Facebook-scale. In *Proceedings of the VLDB Endowment*, Vol. 8. 1804–1815. https://doi.org/10.14778/2824032.2824077

[10] Hanjun Dai, Hui Li, Tian Tian, Xin Huang, Lin Wang, Jun Zhu, and Le Song. 2018. Adversarial Attack on Graph Structured Data. In *International Conference on Machine Learning*, Vol. 80. PMLR, 1115–1124.

[11] Chenhui Deng, Xiuyu Li, Zhuo Feng, and Zhiru Zhang. 2022. GARNET: Reduced-Rank Topology Learning for Robust and Scalable Graph Neural Networks. In *Proceedings of the First Learning on Graphs Conference*. PMLR, 3:1–3:23. https://proceedings.mlr.press/v198/deng22a.html ISSN: 2640-3498.

[12] Chenhui Deng, Xiuyu Li, Zhuo Feng, and Zhiru Zhang. 2022. GARNET: Reduced-Rank Topology Learning for Robust and Scalable Graph Neural Networks. In *Proceedings of the First Learning on Graphs Conference*. PMLR, 3:1–3:23.

[13] Narsingh Deo. 2017. *Graph Theory with Applications to Engineering and Computer Science*. Courier Dover Publications.

[14] Negin Entezari, Saba A. Al-Sayouri, Amirali Darvishzadeh, and Evangelos E. Papalexakis. 2020. 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 (WSDM '20)*. ACM, New York, NY, USA, 169–177. https://doi.org/10.1145/3336191.3371789

[15] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. 2019. Graph Neural Networks for Social Recommendation. In *The World Wide Web Conference (WWW '19)*. Association for Computing Machinery, New York, NY, USA, 417–426. https://doi.org/10.1145/3308558.3313488

[16] Herbert Federer. 2014. *Geometric Measure Theory*. Springer.

[17] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 1025–1035.

[18] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, YongDong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, Virtual Event China, 639–648. https://doi.org/10.1145/3397271.3401063

[19] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the Knowledge in a Neural Network. arXiv:1503.02531 [cs, stat]

[20] Paul Jaccard. 1912. The Distribution of the Flora in tHE Alpine Zone.1. *New Phytologist* 11, 2 (1912). https://doi.org/10.1111/j.1469-8137.1912.tb05611.x

[21] Dejun Jiang, Zhenxing Wu, Chang-Yu Hsieh, Guangyong Chen, Ben Liao, Zhe Wang, Chao Shen, Dongsheng Cao, Jian Wu, and Tingjun Hou. 2021. Could Graph Neural Networks Learn Better Molecular Representation for Drug Discovery? A Comparison Study of Descriptor-Based and Graph-Based Models. *Journal of Cheminformatics* 13, 1 (Feb. 2021), 12. https://doi.org/10.1186/s13321-020-00479-8

[22] Wei Jin, Yao Ma, Xiaorui Liu, Xianfeng Tang, Suhang Wang, and Jiliang Tang. 2020. Graph Structure Learning for Robust Graph Neural Networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '20)*. Association for Computing Machinery, New York, NY, USA, 66–74. https://doi.org/10.1145/3394486.3403049

[23] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations*.

[24] Runlin Lei, Zhen Wang, Yaliang Li, Bolin Ding, and Zhewei Wei. 2022. EvenNet: Ignoring Odd-Hop Neighbors Improves Robustness of Graph Neural Networks. In *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (Eds.), Vol. 35. Curran Associates, Inc., 4694–4706.

[25] Kuan Li, Yang Liu, Xiang Ao, Jianfeng Chi, Jinghua Feng, Hao Yang, and Qing He. 2022. Reliable Representations Make A Stronger Defender: Unsupervised Structure Refinement for Robust GNN. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22)*. Association for Computing Machinery, New York, NY, USA, 925–935. https://doi.org/10.1145/3534678.3539484

[26] Lujun Li and Zhe Jin. 2022. Shadow Knowledge Distillation: Bridging Offline and Online Knowledge Transfer. In *Advances in Neural Information Processing Systems*.

[27] Derek Lim, Felix Hohne, Xiuyu Li, Sijia Linda Huang, Vaishnavi Gupta, Omkar Bhalerao, and Ser Nam Lim. 2021. Large Scale Learning on Non-Homophilous Graphs: New Benchmarks and Strong Simple Methods. In *Advances in Neural Information Processing Systems*, Vol. 34. Curran Associates Inc., 20887–20902.

[28] Yang Liu, Xiang Ao, Zidi Qin, Jianfeng Chi, Jinghua Feng, Hao Yang, and Qing He. 2021. Pick and Choose: A GNN-based Imbalanced Learning Approach for Fraud Detection. In *Proceedings of the Web Conference 2021 (WWW '21)*. Association for Computing Machinery, New York, NY, USA, 3168–3177. https://doi.org/10.1145/3442381.3449989

[29] Yixin Liu, Ming Jin, Shirui Pan, Chuan Zhou, Yu Zheng, Feng Xia, and Philip Yu. 2022. Graph Self-Supervised Learning: A Survey. *IEEE Transactions on Knowledge and Data Engineering* (2022), 1–1. https://doi.org/10.1109/TKDE.2022.3172903

[30] Ilya Loshchilov and Frank Hutter. 2016. SGDR: Stochastic Gradient Descent with Warm Restarts. In *International Conference on Learning Representations*.

[31] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2022. Towards Deep Learning Models Resistant to Adversarial Attacks. In *International Conference on Learning Representations*.

[32] Kenta Oono and Taiji Suzuki. 2020. Graph Neural Networks Exponentially Lose Expressive Power for Node Classification. In *International Conference on Learning Representations*.

[33] Maddy Osman. 2022. Wild and Interesting Facebook Statistics and Facts (2023).

[34] A. Pal, C. Eksombatchai, Y. Zhou, B. Zhao, C. Rosenberg, and J. Leskovec. 2020. Pinnersage: Multi-modal User Embedding Framework for Recommendations at Pinterest. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2311–2320. https://doi.org/10.1145/3394486.3403280

[35] Tianyu Pang, Min Lin, Xiao Yang, Jun Zhu, and Shuicheng Yan. 2022. Robustness and Accuracy Could Be Reconcilable by (Proper) Definition. In *Proceedings of the 39th International Conference on Machine Learning*. PMLR, 17258–17277.

[36] Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. 2020. Geom-GCN: Geometric Graph Convolutional Networks. In *International Conference on Learning Representations*.

[37] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online Learning of Social Representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '14)*. Association for Computing Machinery, New York, NY, USA, 701–710. https://doi.org/10.1145/2623330.2623732

[38] Mary Phuong and Christoph Lampert. 2019. Towards Understanding Knowledge Distillation. In *Proceedings of the 36th International Conference on Machine Learning*. PMLR, 5142–5151.

[39] David Pujol-Perich, Jose Suarez-Varela, Albert Cabellos-Aparicio, and Pere Barlet-Ros. 2022. Unveiling the Potential of Graph Neural Networks for Robust Intrusion Detection. *ACM SIGMETRICS Performance Evaluation Review* 49, 4 (June 2022), 111–117. https://doi.org/10.1145/3543146.3543171

[40] Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. 2020. DropEdge: Towards Deep Graph Convolutional Networks on Node Classification. In *International Conference on Learning Representations*.

[41] Krzysztof Rusek, José Suárez-Varela, Paul Almasan, Pere Barlet-Ros, and Albert Cabellos-Aparicio. 2020. RouteNet: Leveraging Graph Neural Networks for Network Modeling and Optimization in SDN. *IEEE Journal on Selected Areas in Communications* 38, 10 (Oct. 2020), 2260–2270. https://doi.org/10.1109/JSAC.2020.3000405

[42] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. 2008. Collective Classification in Network Data. *AI Magazine* 29, 3 (Sept. 2008), 93–93. https://doi.org/10.1609/aimag.v29i3.2157

[43] Yijun Tian, Chuxu Zhang, Zhichun Guo, Xiangliang Zhang, and Nitesh Chawla. 2023. Learning MLPs on Graphs: A Unified View of Effectiveness, Robustness, and Efficiency. In *International Conference on Learning Representations*.

[44] Fred Tung and Greg Mori. 2019. Similarity-Preserving Knowledge Distillation. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, Seoul, Korea (South), 1365–1374. https://doi.org/10.1109/ICCV.2019.00145

[45] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *International Conference on Learning Representations*.

[46] Petar Veličković, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2019. Deep Graph Infomax. In *International Conference on Learning Representations*.

[47] Aladin Virmaux and Kevin Scaman. 2018. Lipschitz Regularity of Deep Neural Networks: Analysis and Efficient Estimation. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. 3839–3848.

[48] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous Graph Attention Network. In *The World Wide Web Conference*. ACM, San Francisco CA USA, 2022–2032. https://doi.org/10.1145/3308558.3313562

[49] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. Simplifying Graph Convolutional Networks. In *Proceedings of the International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 6861–6871.

[50] Huijun Wu, Chen Wang, Yuriy Tyshetskiy, Andrew Docherty, Kai Lu, and Liming Zhu. 2019. Adversarial Examples for Graph Data: Deep Insights into Attack and Defense. In *Proceedings of the International Joint Conference on Artificial Intelligence*. ijcai.org, 4816–4823.

[51] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. 2021. A Comprehensive Survey on Graph Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems* 32, 1 (Jan. 2021), 4–24,. https://doi.org/10.1109/TNNLS.2020.2978386

[52] Kaidi Xu, Hongge Chen, Sijia Liu, Pin-Yu Chen, Tsui-Wei Weng, Mingyi Hong, and Xue Lin. 2019. Topology Attack and Defense for Graph Neural Networks: An Optimization Perspective. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI'19)*. AAAI Press, Macao, China, 3961–3967.

[53] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. 2018. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. In *Proceedings of the 24th International Conference on Knowledge Discovery & Data Mining*. ACM, London United Kingdom, 974–983. https://doi.org/10.1145/3219819.3219890

[54] Shichang Zhang, Yozen Liu, Yizhou Sun, and Neil Shah. 2022. Graph-Less Neural Networks: Teaching Old MLPs New Tricks via Distillation. In *International Conference on Learning Representations*.

[55] Xiang Zhang and Marinka Zitnik. 2020. GNNGUARD: Defending Graph Neural Networks against Adversarial Attacks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems (NIPS'20)*. Curran Associates Inc., Red Hook, NY, USA, 9263–9275.

[56] D. Zhu, Z. Zhang, P. Cui, and W. Zhu. 2019. Robust Graph Convolutional Networks against Adversarial Attacks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '19)*. Association for Computing Machinery, New York, NY, USA, 1399–1407. https://doi.org/10.1145/3292500.3330851

[57] Jiong Zhu, Junchen Jin, Donald Loveland, Michael T. Schaub, and Danai Koutra. 2022. How Does Heterophily Impact the Robustness of Graph Neural Networks? Theoretical Connections and Practical Implications. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22)*. Association for Computing Machinery, New York, NY, USA, 2637–2647. https://doi.org/10.1145/3534678.3539418

[58] Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. 2020. Beyond Homophily in Graph Neural Networks: Current Limitations and Effective Designs. In *Advances in Neural Information Processing Systems*, Vol. 33. Curran Associates, Inc., 7793–7804.

[59] R. Zhu, K. Zhao, H. Yang, W. Lin, C. Zhou, B. Ai, Y. Li, and J. Zhou. 2019. Aligraph: A Comprehensive Graph Neural Network Platform. *Proceedings of the VLDB Endowment* 12, 12 (2019), 2094–2105. https://doi.org/10.14778/3352063.3352127

[60] Daniel Zügner, Amir Akbarnejad, and Stephan Günnemann. 2018. Adversarial Attacks on Neural Networks for Graph Data. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, London United Kingdom, 2847–2856. https://doi.org/10.1145/3219819.3220078

[61] Daniel Zügner and Stephan Günnemann. 2019. Adversarial Attacks on Graph Neural Networks via Meta Learning. In *International Conference on Learning Representations*.