Front. Comput. Sci., 2025, 19(10): 1910359
https://doi.org/10.1007/s11704-024-40533-4

RESEARCH ARTICLE

Common knowledge learning for generating transferable adversarial examples

Ruijie YANG[1], Yuanfang GUO (✉)[1], Junfu WANG[1], Jiantao ZHOU[2], Yunhong WANG[1]

1 Laboratory of Intelligent Recognition and Image Processing, School of Computer Science and Engineering, Beihang University, Beijing 100191, China

2 The State Key Laboratory of Internet of Things for Smart City and the Department of Computer and Science, University of Macau, Macao 999078, China

Abstract This paper focuses on an important type of black-box attacks, i.e., transfer-based adversarial attacks, where the adversary generates adversarial examples using a substitute (source) model and utilizes them to attack an unseen target model, without knowing its information. Existing methods tend to give unsatisfactory adversarial transferability when the source and target models are from different types of DNN architectures (e.g., ResNet-18 and Swin Transformer). In this paper, we observe that the above phenomenon is induced by the output inconsistency problem. To alleviate this problem while effectively utilizing the existing DNN models, we propose a common knowledge learning (CKL) framework to learn better network weights to generate adversarial examples with better transferability, under fixed network architectures. Specifically, to reduce the model-specific features and obtain better output distributions, we construct a multi-teacher framework, where the knowledge is distilled from different teacher architectures into one student network. By considering that the gradient of input is usually utilized to generate adversarial examples, we impose constraints on the gradients between the student and teacher models, to further alleviate the output inconsistency problem and enhance the adversarial transferability. Extensive experiments demonstrate that our proposed work can significantly improve the adversarial transferability.

Keywords black-box attack, adversarial transferability, deep neural networks

1 Introduction

Deep neural networks (DNNs), such as convolutional neural networks (CNNs) [1–3] and transformers [4,5], have demonstrated significant advancements across various machine learning tasks [6–10]. However, recent studies reveal that DNNs are vulnerable to adversarial attacks [11–13], which add special yet imperceptible perturbations to benign data to deceive the deep models into making wrong predictions. This vulnerability poses a considerable threat to the safety of the DNN-based systems [14–16], especially those applied in security-sensitive domains, such as autonomous driving and face-scan payment.

Since different DNN architectures usually function differently, their corresponding vulnerabilities are also different. Therefore, existing adversarial attack techniques are usually specifically designed for different DNN architectures, to discover the safety threats of different DNN architectures. Due to privacy or copyright protection considerations, black-box attacks tend to possess more applicability in real scenarios. In this paper, we focus on an extensively studied scenario of black-box attacks, i.e., transfer-based adversarial attack, which assumes that the adversary does not have access to the target model. Instead, the attacker can train substitute models to generate adversarial examples to attack the target model.

For common CNN models, to enhance the transferability of adversarial examples, various techniques have been proposed, which can be briefly classified into two categories according to their mechanisms, i.e., gradient modifications [12,14,17] and input transformations [18–21]. The former type of methods usually improves the gradient ascend process for adversarial attacks to prevent the adversarial examples from over-fitting to the source model. The latter type of methods usually manipulates input images with various transformations, which enables the generated adversarial perturbations to adapt different input transformations. Consequently, these adversarial examples possess a higher probability of transferring to successfully attack the target unknown model.

The recent success of vision transformers (ViT) has also prompted several studies on devising successful attacks on the ViT type of architectures. [22–24] construct substitute model based attack methods for ViTs, according to their unique architectures, such as attention module, classification token, to generate transferable adversarial examples.

Currently, existing methods usually employ pre-trained classification models as the source (substitute) model (as well

Received May 26, 2024; accepted December 15, 2024

E-mail: andyguo@buaa.edu.cn

as the target model in the experiments) directly, for achieving transfer-based adversarial attacks. One of the core reasons, which affects the transferability of these adversarial attacks, is the similarity between the source (substitute) model and the target model. By assuming that the source model and the target model are identical, the attack becomes a white-box attack. Then, the transferability is expected to be high, and the attack success rate is equivalent to that in the corresponding white-box setting. Intuitively, two models with similar architectures and similar weights tend to possess high transferability [25]. On the contrary, models with significantly different architectures and weights usually exhibit low transferability. For example, when we generate adversarial examples in ResNet-18 and test them on ViT-S, the attack success rate is 45.99%, which is lower than the transferability from ResNet-18 to Inception-v3 (62.87%).

Since different network architectures and weights will induce different outputs, we believe that the low adversarial transferability is caused by the *output inconsistency* problem. As can be observed in Fig. 1, even when each of the models gives correct classification result, the output probabilities are still inconsistent. Besides, the inconsistency between two different CNN models is usually smaller than that between two models from different architectural categories, e.g., a CNN model and a transformer-based model. Apparently, this inconsistency is harmful to adversarial transferability, because adversarial attacks are usually designed to manipulate the target model's output probability and this inconsistency will increase the uncertainty of the outputs of the target model. To better describe this output inconsistency, KL divergence is employed to numerically represent it. As shown in Fig. 2, higher output inconsistency tends to induce lower transferability and vice versa.

To alleviate the above problem, a straightforward solution is to construct a universal network architecture which possesses relatively similar output distributions as different types of DNN models. Unfortunately, this universal network architecture and its training strategy are both difficult to be designed and implemented. Besides, this solution is highly unlikely to effectively utilize the existing pre-defined DNN architectures, which are much more convenient to be applied in real scenarios.

To alleviate this *output inconsistency* problem and effectively utilize the existing pre-defined DNN models, in this paper, we propose a common knowledge learning (CKL) method for the substitute (source) model to learn better network weights to generate adversarial examples with better transferability, under fixed network architectures. Specifically, to reduce the model-specific features and obtain better output distributions, we adopt a multi-teacher approach, where the knowledge is distilled from different teacher architectures into one student network. By considering that the gradient of input is usually utilized to generate adversarial examples, we impose constraints on the gradients between the student and teacher models. Since multiple teach models may generate conflicting gradients, which will interfere the optimization process, motivated by PCGrad [26], we introduce the model gradients constraint term into our work to diminish the gradient conflicts of the teacher models.

Our contributions are summarized as follow.

- We analyze the relationship between adversarial transferability and property of the substitute (source) model, and observe that a substitute model with less output inconsistency to the target model tends to possess better adversarial transferability.

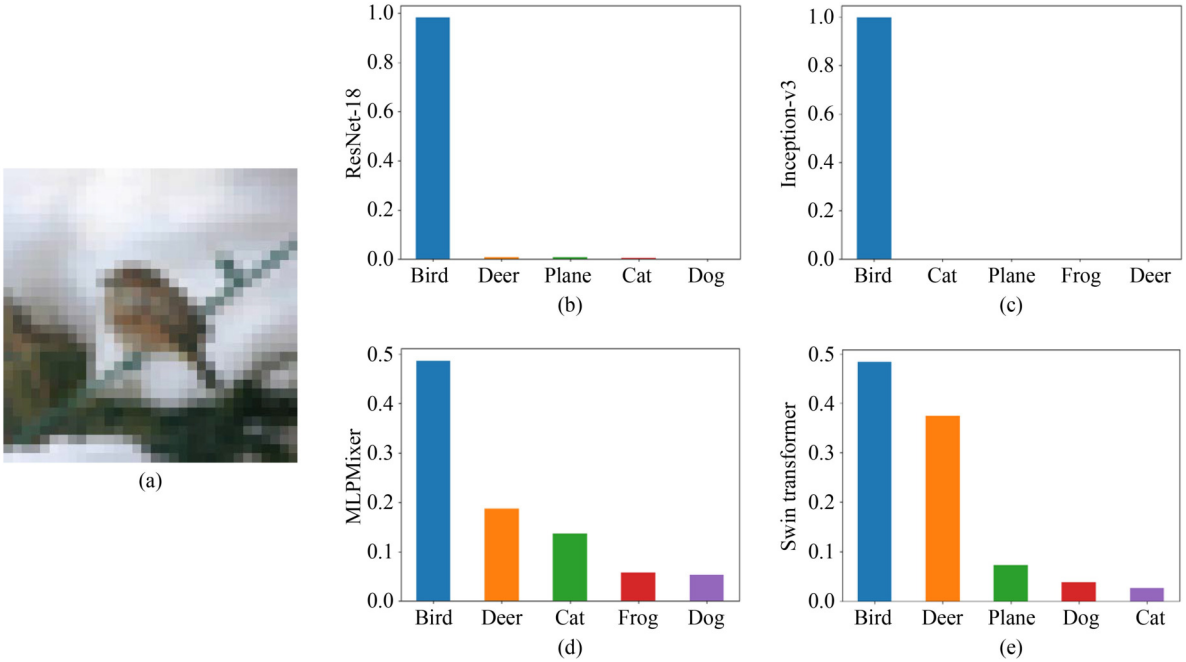


Fig. 1 Output inconsistency among different networks with the same input image, whose truth label is 'bird'. (a) is a randomly selected input image from CIFAR-10, and (b)−(e) denote the output confidences of different models. Although every model gives the correct prediction, the output probabilities are obviously different. The output discrepancy leads to the poor transferability of adversarial examples

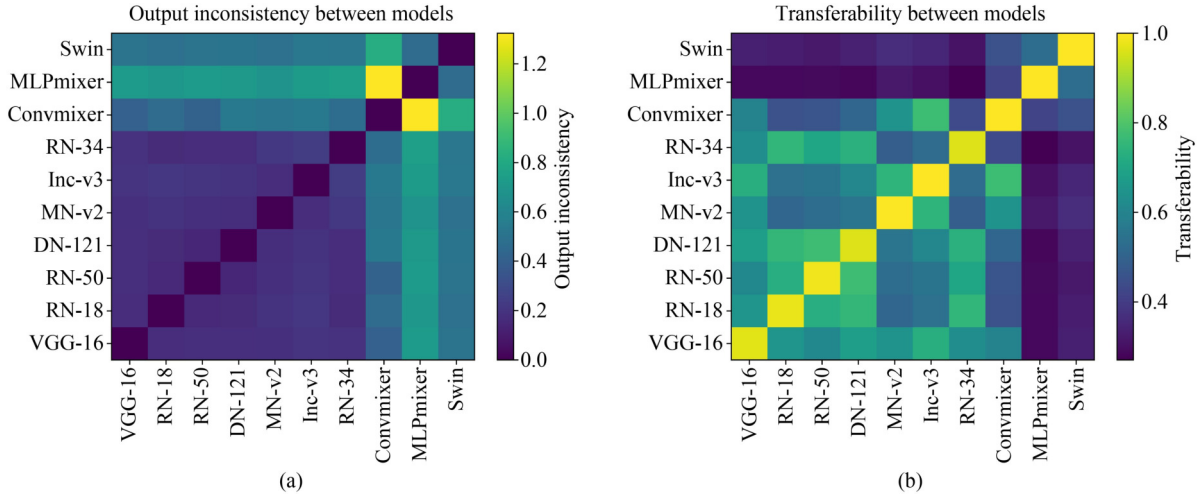


Fig. 2 Output inconsistency and transferability on the CIFAR-10 testing set. The results are obtained by averaging from 10,000 images. (a) The output inconsistency. We calculate the KL divergence of different model outputs on the same input image, to represent the output inconsistency. A higher value denotes a higher inconsistency; (b) We take two models in turn as the source model to generate adversarial examples to attack the other and compute the transferability by averaging the two attack results

- To reduce the model-specific features and obtain better output distributions, we propose a common knowledge learning framework to distill multi-teacher knowledge into one single student network.
- For generating adversarial examples with better transferability, we propose to learn the input gradients of the teacher models and utilize gradient projection to reduce the conflicts in the gradients of multiple teachers.
- Extensive experiments on CIFAR-10, CIFAR-100, and TinyImageNet demonstrate that our method is effective and can be easily integrated into transfer-based adversarial attack methods to significantly improve their attack performances.

2 Related work

2.1 Adversarial attacks

Adversarial attack is firstly proposed by [27]. Subsequently, a large number of adversarial attack methods are proposed, which are usually classified into two categories according to the adversary's knowledge [28], i.e., white-box and black-box attacks. The black-box attacks can be further classified into query-based and transfer-based attacks. White-box attacks usually assume that the adversary can access all the necessary information, including the architecture, parameters and training strategy, of the target model. Query-based attacks usually assume that the adversary can obtain the outputs by querying the target model [29–33]. Transfer-based attacks generate adversarial examples without the access to the target model, whose assumption is the closest to practice. Under such circumstance, the adversary usually exploits a substitute model to generate adversarial examples and utilize the examples to deceive the target model [18,19,22,34–36]. Since our work focus on the transfer-based scenario, we will introduce this attack scenario in details in next subsection.

2.2 Transfer-based attacks

Since different DNN architectures usually function differently, existing transfer-based attack techniques are usually specifically designed for different DNN architectures.

For CNN architectures, the attack approaches in transfer-based scenarios can mainly be classified into two categories, i.e., gradient modifications and input transformations. For gradient modifications based methods, [14] firstly proposes MI-FGSM to stabilize the update directions with a momentum term to improve the transferability of adversarial examples. [12] adopts the Nesterov accelerated gradients into the iterative attacks. [17] proposes an Adam iterative fast gradient tanh method (AI-FGSM) to generate adversarial examples with high transferability. Besides, [37] adopts the AdaBelief optimizer into the update of the gradients and constructs ABI-FGM to further boost the attack success rates of adversarial examples. Recently, [38] introduces variance tuning to further enhance the adversarial transferability of iterative gradient-based attack methods.

On the contrary, input transformations based methods usually applies various transformations to the input image in each iteration to prevent the attack method from being overfitting to the substitute model. [19] presents a translation-invariant attack method, named TIM, by optimizing a perturbation over an ensemble of translated images. Inspired by data augmentations, [37] optimizes adversarial examples by adding image cropping operation to each iteration of the perturbation generation process. Recently, Admix [20] calculates the gradient on the input image admixed with a small portion of each add-in image while using the original label, to craft adversarial examples with better transferability. [21] improves adversarial transferability via an adversarial transformation network, which studies efficient image transformations to boosting the transferability. [39] proposes AutoMa to seek for a strong model augmentation policy based on reinforcement learning.

Since the above approaches are designed for CNNs, their performances degrade when their generated adversarial examples are directly input to other types of DNN architectures, such as vision transformers [4] and mlpmixer

[40]. Since the transformer-based architectures have also been widely applied in image classification task, several literatures have also presented transfer-based adversarial attack methods when transformer-based architectures are employed as the source (substitute) model. [22] proposes a self-attention gradient attack (SAGA) to enhance the adversarial transferability. [23] introduces two novel strategies, i.e., self-ensemble and token refinement, to improve the adversarial transferability of vision transformers. Motivated by the observation that the gradients of attention in each head impair the generation of highly transferable adversarial examples, [24] presents a pay no attention (PNA) attack, which ignores the backpropagated gradient from the attention branch.

2.3 Knowledge distillation

A common technique for transferring knowledge from one model to another is knowledge distillation. The mainstream knowledge distillation algorithms can be classified into three categories, i.e., response-based, feature-based and relation-based methods [41]. The feature-based methods [42,43] exploit the outputs of intermediate layers in the teacher model to supervise the training of the student model. The relation-based methods [44] explore the relationships between different layers or data samples. These two types of methods requires synchronized layers in both the teacher and student models. However, when the architectures of the teacher and student models are inconsistent, the selection of proper synchronized layers becomes difficult to achieve. On the contrary, [45], which is a response-based method, constrains the logits layers of the teacher and student models, which can be easily implemented for different tasks without the above mentioned synchronization problem. Therefore, [45] is adopted in our proposed work.

3 Methodology

3.1 Notations

Here, we define the notations which will be utilized in the rest of this paper. Let $x \in \mathcal{X} \subseteq R^{C\times W\times H}$ denote a clean image and y is its corresponding label, where $\mathcal{X}$ is the image space. Let $z = (o_1, o_2, ..., o_i, ..., o_K) \in R^K$ be the output logit of a DNN model, where K is the number of classes. $T_1(\cdot), T_2(\cdot), ..., T_N(\cdot)$ denote the teacher networks and N is the total number of teacher models. $S(\cdot)$ stands for the student network. $L(S(x), y)$ is utilized as the loss function, e.g., cross-entropy loss, respect to input image x, model S and label y. For simplicity, we also use L_S, L_{T_i} to denote $L(S(x), y)$ and $L(T_i(x), y)$, respectively, in the rest section. The goal of an generated adversarial example x^* is to deceive the target DNN model, such that the prediction result of the target model $F_{tar}(\cdot)$ is not y, i.e., $\underset{i}{argmax}\, F_{tar}(x^*) \neq y$. Meanwhile, the adversarial example is usually desired to be similar to the original input, which is usually achieved by constraining the adversarial perturbation by L_p norm, i.e., $\|x^* - x\|_p \leqslant \epsilon$, where ϵ is a predefined small constant.

3.2 Overview

According to Figs. 1 and 2, the output inconsistency problem significantly affects the transferability of adversarial examples, i.e., high output inconsistency usually indicates low transferability, and vice versa. Since the output inconsistency within each type of DNN architectures is usually less than that of the cross-architecture models, the adversarial examples generated based on the substitute model from one type of DNN architectures (e.g., CNNs) usually give relatively poor attack performance on the target models from other types of DNN architectures (e.g., ViT, MLPMixer).

To alleviate the above problems, in this paper, we propose a common knowledge learning (CKL) framework, which distills the common knowledge of multiple teacher models with different architectures into a single student model, to obtain better substitute models. The overall framework is shown in Fig. 3. Firstly, we select teacher models from different types of DNN architectures. The student model will learn from their outputs to reduce the model-specific features and obtain common (model-agnostic) features, to alleviate the output inconsistency problem. Since the input gradient is always utilized in typical adversarial attack process, we also design a

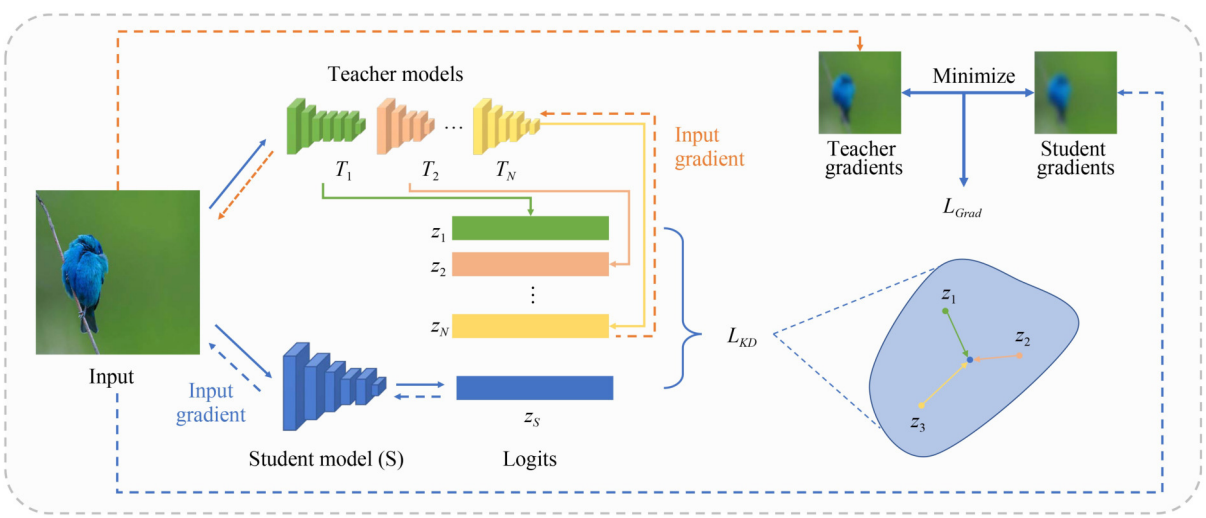


Fig. 3 Our common knowledge learning (CKL) framework. We leverage the input gradient distillation loss and knowledge distillation loss to force the student model (S) to learn the common knowledge from multiple teacher models

constraint on the input gradients between the teacher models and the student model, to further promote the transferability of generated adversarial examples. After training the student model, in the testing stage, this model will be utilized as the source (substitute) model to generate adversarial examples.

3.3 Common knowledge distillation

As can be observed from Fig. 1, when the same input is fed into different models, the output probabilities are quite different, which actually reveals that there exists feature preference in deep models. Apparently, the output inconsistency problem is induced by these model-specific features, because these features, which are considered to be distinctive to one model, may not be distinctive enough to others. Under such circumstance, when an adversarial example is misclassified by the source model to the other class $\hat{y}$ ($\hat{y} \neq y$), it contains certain manipulated features which are distinctive to the source model. However, if these manipulated features are not distinctive enough to the target model, the adversarial example may not be able to deceive the target model.

According to the analysis above, it is vital to identify and emphasize the common (model-agnostic) features among different DNN models in the substitute model, such that when these model-agnostic features is manipulated to generate the adversarial examples, the target model will possess a higher possibility to be deceived, i.e., the adversarial transferability will be improved. Therefore, we construct a multi-teacher knowledge distillation method to force the student model to learn and emphasize the common features from various DNN models. Since different DNN models usually possess different architectures, we constrain the model outputs between the student and teacher models, by adopting [45]. Specifically, we employ KL divergence to measure the output discrepancy between the student model $S(\cdot)$ and each teacher model $T_i(\cdot)$. We denote the output of student model as $S(x) = (o_1^S, o_2^S, ..., o_K^S)$ and the output of teacher model as $T_i(x) = (o_1^{T_i}, o_2^{T_i}, ..., o_K^{T_i})$. Then, KL divergence is formulated as

$$KL(S(x), T_i(x)) = \sum_{k=1}^{K} o_k^S \cdot \log \frac{o_k^S}{o_k^{T_i}}, \tag{1}$$

where K represents the number of classes.

To jointly utilize all the teacher models, the knowledge distillation (KD) loss L_{KD} is defined as

$$L_{KD} = \Sigma_{i=1}^{N} KL(S(x), T_i(x)). \tag{2}$$

3.4 Gradient distillation

Since the input gradients are commonly utilized in the mainstream adversarial attack methods, such as FGSM [11], MI-FGSM [14], DI-FGSM [18], TI-FGSM [18], and VNI-FGSM [38], if two networks $F(x)$ and $G(x)$ satisfy $\nabla_x L_F(x) = \nabla_x L_G(x)$, adversarial examples generated by these methods will be identical when either of the two networks are employed as the source model. Additionally, if $\nabla_x L_F(x) = \nabla_x L_G(x)$ for all x, the losses of $F(x)$ and $G(x)$ differ by at most one constant, i.e., there exists a constant C that $L_F(x) = L_G(x) + C$.[1] Since the outputs of two models are more likely to be inconsistent when their losses are different, if the input gradients of two models are similar, these two models tend to possess less output inconsistency. Although this assumption cannot be exactly satisfied in real scenarios, it can still be useful in generating transferable adversarial examples. Therefore, we make the student model to learn the input gradients from the teacher networks, to further improve the adversarial transferability.

As our framework will utilize multiple teacher networks to teach one student model, it is vital to design suitable approaches to learn multiple input gradients. Under such circumstance, a simple solution is to design a multi-objective optimization problem which minimizes the distances between the input gradients of the student model and each teacher model. This optimization problem can be simplified by minimizing the distance between the input gradient of the student model and the averaged input gradients of the teacher models (which can be regarded as a representative value among all the input gradients of the teacher models), as

$$\min \|\nabla_x L_S(x) - \bar{g}(x)\|_2^2, \tag{3}$$

where $\bar{g}(x) = \Sigma_{i=1}^{N} g_i(x)$ and $g_i(x) = \nabla_x L_{T_i}(x)$. For convenience, we let g_i denote $g_i(x)$ and let $\bar{g}$ denote $\bar{g}(x)$ in the rest of this section, when the input x is not necessary to be emphasized.

Unfortunately, there usually exists conflicting gradients among them, which is similar to the gradient conflict problem [26,46] in multi-task learning. This gradient conflict problem means that there exists a j, satisfying $g_j \cdot \bar{g} < 0$. If $\nabla_x L_S(x)$ is gradually closer to $\bar{g}$, this gradient conflict problem will actually move $\nabla_x L_S(x)$ further away from g_j, which is the input gradient of the jth teacher model.

To address this issue, inspired by PCGrad [26] method, we adjust our optimization objective function by projecting one of the two conflicting gradients onto the normal plane of the other gradient. Specifically, when there exist two conflicting gradients g_i and g_j, i.e., $g_i \cdot g_j < 0$, g_i will be replaced by

$$g_i = g_i - \frac{g_i \cdot g_j}{g_j \cdot g_j} g_j \tag{4}$$

to decrease the conflicts among them. This replacement step is performed for all the gradients.[2] After the replacement step, we calculate the averaged value of replaced gradient g_i, i.e., $d(x) = \Sigma_{i=1}^{N} g_i(x)$, for the student model to learn. Then, the gradient loss function becomes

$$L_{Grad} = \|\nabla_x L_S(x) - d(x)\|_2^2. \tag{5}$$

By combining Eqs. (2) and (5), the final loss of our common knowledge learning (CKL) for training the student network can be obtained as

---

[1] This is because that if we set $\psi(x) = L_F(x) - L_G(x)$, $\nabla_x \psi(x) = 0$ satisfies everywhere, which induces that $\psi(x) = C$ is a constant function.

[2] In practice, giving a gradient set $\mathcal{A} = \{g_1, g_2, ..., g_i, ..., g_n\}$, for each g_i, g_j is randomly sampled without replacement in the rest set, i.e., $\mathcal{A} \backslash \{g_i\}$. If there is a conflict between g_i and g_j, we replace g_i by Eq. (4).

Algorithm 1 Process of CKL for one mini-batch

Input: mini-batch (x, y), the teacher models $F_{T_1}, ..., F_{T_N}$, the student model S

Parameter: λ, lr

Get the teacher output $o^{T_i} = T_i(x)$
Get the student output $o^S = S(x)$
Get L_{KD} by Eq. (2)
Calculate the teacher gradient $g_i = \nabla_x L_{T_i}(x)$
Gradient projection by Eq. (4)
Calculate the student gradient $\nabla_x L_S(x)$
Get L_{Grad} by Eq. (5)
Total loss $L_{total} = L_{KD} + \lambda \cdot L_{Grad}$
Update the parameter of S by $w_S = w_S - lr \cdot \nabla_{w_S} L_{total}$

$$L_{total} = L_{KD} + \lambda \cdot L_{Grad}, \tag{6}$$

where λ is the hyperparameter employed to balance the two loss terms, which is further discussed in Section 4.6. The detailed process of our CKL method for one mini-batch is summarized in Algorithm 1.

3.5 Generating adversarial examples with CKL

After the training process, we utilize the trained student model (S) as the source (substitute) model to generate adversarial examples. Our framework can be easily combined with the existing transfer-based adversarial attack methods. For demonstration, here we leverage MI-FGSM [14] as an example to explain the adversarial example generation process. Let $CE(\cdot)$ denote the commonly utilized Cross Entropy loss. We set $x_0 = x$. Then, the adversarial example generation process can be formulated as

$$\begin{aligned} L(x_t) &= CE(S(x_t), y), \\ g_t &= \nabla_{x_t} L(x_t), \\ m_{t+1} &= \mu m_t + \frac{g_t}{\|g_t\|_1}, \\ x_{t+1} &= Clip_{x,\epsilon}(x_t + \alpha \cdot sign(m_{t+1})), \end{aligned} \tag{7}$$

where ϵ is a predefined small constant to constrain the maximum magnitudes of the generated adversarial example. $Clip_{x,\epsilon}(\cdot)$ forces the modified value to stay inside the L_∞ ball $(\{x_t \| \|x_t - x\|_\infty \leqslant \epsilon\})$. α is the step size and μ is momentum. This process terminates when t reaches the maximum number (N) of iterations, and x_N is the finally generated adversarial example.

4 Experiments

In this section, we firstly introduce the experimental settings in Section 4.1. Next, we present non-targeted attack experiments in Section 4.2, and targeted attack experiments in Section 4.3. Then, we give the experimental results compared to ensemble attack and intermediate-based attack, in Section 4.4 and Section 4.5. At last, we conduct ablation studies to evaluate the effectiveness of our proposed modules in Section 4.6.

4.1 Experimental settings

Datasets We consider three widely used classification datasets, i.e., the CIFAR-10, CIFAR-100 [47], and TinyImageNet, in our experiments. CIFAR-10 and CIFAR-100 datasets possess images with the size of $32 \times 32 \times 3$. The numbers of classes in CIFAR-10 and CIFAR-100 are 10 and 100, respectively. For these two CIFAR datasets, we utilize the official training and testing sets, which respectively contains 50000 images and 10000 images, to train the student models and generate adversarial examples. TinyImageNet, in which the number of classes is 200, possesses 100000 images in its official training set and 10000 images in its official validation set, with the image size of $64 \times 64 \times 3$. For TinyImageNet, we employ its training set to train the student models and utilize its validation set to generate adversarial examples. Since the widely used DNNs usually give a low accuracy on TinyImageNet and it is not quite reasonable to attack the samples which are already misclassified by DNNs, we select 1000 images that are correctly classified by all the target models to produce adversarial examples.

Networks Nine networks with different types of DNN architectures are employed as either the source model or the target model, which includes ResNets [3], VGG-16 [2], DenseNet-121 [48], Inception-v3 [49], MobileNet-v2 [50], ViT-S [4], Swin Transformers [5], MLPMixer [40], and ConvMixer [51]. To learn common knowledge of neural networks, we select teacher models from different types of DNN architectures. Thus, ResNet-50, Inception-v3, Swin-T, and MLPMixer are constructed as the teacher models in the subsequent experiments.

Baselines Five baselines are employed in our work, including non-targeted attack, i.e., MI-FGSM [14], DI-FGSM [18], VNI-FGSM [38], ILA-DA [52], and targeted attack, Logit [53]. For non-targeted attack, the number of attack iterations M is set to 30 and step size is set to $1/255$. For targeted attack, following the experimental settings in [53], M is set to 300 and step size is set to $2/255$.

Implementation details We employ the training set to train the student model (S) and testing set for generating the adversarial examples. In the training process, the momentum SGD optimizer is employed, with an initial learning rate $lr = 0.1$ (annealed down to zero following a cosine schedule), momentum 0.9, and weight decay 0.0003. The maximum epoch number is 600. In the attack stage, we constrain the adversarial example x^* and origin input x by the l_∞ ball with $\epsilon = 8/255$, i.e., $\|x^* - x\|_\infty \leqslant 8/255$. For DI-FGSM [18], each input benign image is randomly resized to $rnd \times rnd$, with $rnd \in [28, 32)$ ($[56, 64)$ for TinyImageNet), and then padded to the size 32×32 (64×64 for TinyImageNet) in a random manner.

Evaluation metric The attack success rate (ASR) is employed to evaluate the attack performance. It is defined as the probability that the target model is fooled by the generated adversarial examples, i.e.,

$$ASR = 1 - \frac{\#\{correct\ samples\}}{\#\{total\ samples\}}, \tag{8}$$

where # denotes the number of elements in the set.

4.2 Non-targeted attack evaluation

4.2.1 CIFAR10 results

The attack success rates (ASR) of the non-targeted attack on CIFAR-10 are reported in Table 1. We compare our method with MI-FGSM, DI-FGSM, and VNI-FGSM, which are abbreviated as MI, DI, and NI, respectively. Their original methods generate adversarial examples by directly employing the pre-trained models. Meanwhile, our adversarial examples are generated by the student model, which is trained by our CKL framework. As can be observed, expect for the white-box attacks, i.e., the source model and target model are same, our CKL can give significant improvements compared to the corresponding baseline methods, which proves the effectiveness of our proposed work for adversarial transferability. Since the transferability to the unseen models is more consistent with transfer attack scenario, we calculate the averaged ASR value of unseen models (other models) in the last column, which do not overlap with teacher models.

Transferability to the unseen models Note that ResNet-50, Inception-v3, Swin Transformer, and, MLPMixer are employed as teacher models and utilized to train the student models. As can be observed, when selected these four models as the target models, the results with our CKL framework are significantly improved, compared to their corresponding baselines. To better verify the effectiveness of our CKL, we also employ the unseen models, e.g., VGG-16, DenseNet-121, Convmixer and ViT-S, as target models for evaluation, and our CKL can also achieve significant improvements. We also compute the averaged ASR value of the unseen models in the last column. We can observe that our method have consistent improvement on all baselines and substitute models. Our CKL method can improve the averaged value for **at least 10 percentage points**, which indicates that our CKL framework can learn actually effective common knowledge from the teacher models and leverage them to the unseen models.

Transferability to the cross-architecture models The cross-architecture transferability is usually a challenging problem for the baseline attack methods, as can be observed from the results. For example, when the source model is selected as ResNet-18, the correspondingly generated adversarial examples' transferability to ViT-S is relatively low, i.e., only 37.16% for MI-FGSM. On the contrary, by integrating our CKL framework, the attack transferability can be largely improved (62.14% for MI-FGSM-CKL), which benefits from the common knowledge learned at the training stage.

4.2.2 CIFAR100 results

For better assessment of our proposed work, we further validate our CKL method on CIFAR-100 and the results are shown in Table 2. The experimental setups are identical to these in Section 4.1. MI-FGSM, DI-FGSM, and VNI-FGSM are employed as the baseline methods.

As can be observed, our method has a consistent improvement on the CIFAR-100 dataset, whatever the attack method and the source model are. Besides, we compute the averaged ASRs of the unseen models in the last column of Table 2. From the results, we can observe that our CKL method has a decent improvement, compared to the corresponding baseline methods in different substitute models. In addition, for the cross-architecture transferability, our method usually gives an improvement of about 10 percentage points. For example, when the source model is ResNet-18 and the target model is ConvMixer, 'MI+CKL' outperforms 'MI' up to 10.86% and 'DI+CKL' outperforms 'DI' up to 11.93%.

Table 1 Non-targeted attack success rates (%) on CIFAR-10. The first column introduces the source models and the second row presents the target models. The third row gives the clean accuracy of these target models. We report the averaged attack success rate on the entire CIFAR-10 testing set. $^{\triangle}$ implies that the source and target model s are identical. MI-FGSM, DI-FGSM and VNI-FGSM are abbreviated as 'MI', 'DI', 'VNI', respectively. '+CKL' represents that our CKL framework is integrated

	Method	Teacher models				Other models				
		ResNet-50	Inception-v3	Swin-T	MLPMixer	VGG-16	DenseNet-121	ConvMixer	ViT-S	Avg.*
Clean accuracy		93.87	95.34	85.55	78.93	94.02	94.56	91.36	78.74	
ResNet-18	MI [14]	70.67	62.82	45.99	39.17	73.73	74.75	58.22	37.16	60.97
	MI+CKL	86.30	83.09	82.02	71.78	87.66	88.37	81.71	62.14	**79.97(+19.00)**
	DI [18]	77.84	69.39	56.17	44.80	79.87	83.00	65.20	43.00	67.77
	DI+CKL	93.17	89.75	90.07	81.57	93.84	94.86	89.41	73.51	**87.91(+20.14)**
	VNI [38]	76.34	71.61	53.78	43.56	81.39	81.87	68.69	40.99	68.34
	VNI+CKL	91.82	90.30	90.79	82.97	93.77	94.09	90.97	72.50	**87.83(+19.59)**
ResNet-50	MI [14]	98.38$^{\triangle}$	66.89	47.36	41.60	72.76	78.43	61.82	38.90	62.98
	MI+CKL	89.11	85.69	80.11	68.64	86.90	87.97	83.66	58.93	**79.37(+16.39)**
	DI [18]	99.86$^{\triangle}$	77.60	58.32	48.22	82.82	89.18	71.44	46.43	72.47
	DI+CKL	95.61	92.59	88.83	79.14	93.33	95.16	91.12	70.82	**87.61(+15.14)**
	VNI [38]	99.69$^{\triangle}$	78.88	56.91	47.08	83.08	88.33	75.65	43.65	72.68
	VNI+CKL	95.23	92.89	90.05	80.98	93.60	94.57	92.85	70.75	**87.94(+15.26)**
Swin-T	MI [14]	16.39	25.45	100$^{\triangle}$	48.17	23.49	17.23	43.61	38.43	30.69
	MI+CKL	26.25	39.02	99.79	78.16	37.28	27.64	59.12	55.10	**44.79(+10.10)**
	DI [18]	25.98	35.75	100$^{\triangle}$	57.38	35.17	28.23	56.25	49.37	42.26
	DI+CKL	37.28	52.53	99.86	83.94	50.37	40.68	72.27	66.27	**57.40(+15.14)**
	VNI [38]	16.11	27.01	100$^{\triangle}$	51.68	24.75	16.52	48.42	40.03	32.43
	VNI+CKL	27.26	43.34	99.94	84.68	40.87	29.05	66.90	60.48	**49.33(+16.90)**

* 'Avg.' denotes the average ASR value of unseen models (Other models), not overlapping with teacher models.

Table 2 Non-targeted attack success rates (%) on CIFAR-100. The first column introduces the source models and the second row presents the target models. The second column gives the attack methods. The third row gives the clean accuracy of these target models. We report the averaged attack success rate on the entire testing set. MI-FGSM, DI-FGSM, and VNI-FGSM are abbreviated as 'MI', 'DI', 'VNI', respectively. '+CKL' represents that our CKL framework is integrated

	Method	Teacher models				Other models				
		ResNet-50	Inception-v3	Swin-T	MLPMixer	VGG-16	DenseNet-121	ConvMixer	ViT-S	Avg.
Clean accuracy		75.20	75.67	52.07	58.98	74.56	77.00	75.84	43.12	
ResNet-18	MI [14]	79.90	69.65	64.06	60.46	84.08	80.16	68.15	64.51	74.23
	MI+CKL	87.87	87.19	76.59	76.28	91.32	86.49	79.01	71.12	**81.99(+7.76)**
	DI [18]	84.83	76.19	68.14	65.16	88.97	85.62	72.79	67.37	78.69
	DI+CKL	91.82	92.35	81.69	82.47	95.05	91.47	84.72	75.86	**86.78(+8.09)**
	VNI [38]	84.81	76.41	67.75	64.45	89.69	85.48	79.32	66.35	80.21
	VNI+CKL	91.37	92.12	82.09	82.14	95.03	90.90	84.25	74.54	**86.18(+5.97)**
ResNet-50	MI [14]	97.57$^{\triangle}$	69.24	64.65	60.80	79.57	77.47	69.21	64.19	72.61
	MI+CKL	91.46	89.49	77.98	78.47	90.80	87.95	81.99	72.10	**83.21(+10.60)**
	DI [18]	99.48$^{\triangle}$	77.52	69.83	66.85	86.90	85.76	76.55	67.58	79.20
	DI+CKL	94.38	93.91	83.75	85.04	94.47	92.44	86.98	77.43	**87.83(+8.63)**
	VNI [38]	99.34$^{\triangle}$	77.62	70.22	66.33	87.46	84.27	77.15	66.33	78.80
	VNI+CKL	94.11	94.22	84.15	84.70	94.99	92.07	87.47	75.81	**87.59(+8.79)**
Swin-T	MI [14]	43.16	50.28	100$^{\triangle}$	75.40	53.14	41.76	61.39	73.62	57.48
	MI+CKL	48.16	60.54	82.38	79.57	59.04	48.01	64.76	68.71	**60.13(+2.65)**
	DI [18]	49.63	57.57	99.99$^{\triangle}$	79.24	57.87	48.88	65.27	79.31	62.83
	DI+CKL	57.25	73.44	85.59	83.48	68.90	59.22	72.79	74.67	**68.89(+6.06)**
	VNI [38]	42.77	49.59	100$^{\triangle}$	73.53	52.78	41.10	61.94	71.98	56.95
	VNI+CKL	51.31	64.19	85.47	82.74	62.76	51.26	68.88	70.48	**63.35(+6.40)**

These results further verify the effectiveness of our CKL method.

4.2.3 TinyImageNet results

We also conduct non-targeted adversarial attack experiments on TinyImageNet, to validate the effectiveness of our CKL method. Considering that some models perform poorly on TinyImageNet, we choose VGG-16, ResNets, DenseNet-121, ViT-S, Swin-T, Convmixer in this experiment. ResNet-50 and ViT-S are employed as teacher models and others are employed as source or target models. According to the strategy presented in Section 4.1, we randomly select 1,000 testing samples which are all correctly classified by the networks in the non-targeted attack experiments. The results are shown in Table 3.

As can be observed, the attack success rate of baseline method is relatively low. After applying our CKL method, which enables the substitute model to learn common knowledge from different architectures, the generated adversarial examples usually possess high ASR. We point that even attacking unseen models, our CKL method also gives a significant improvement. We also compute the averaged attack success rates of unseen models in the last column. As can be observed, our method improves the averaged ASR values of unseen models for **at least 20 percentage points**, compared to the baselines.

4.3 Targeted attack evaluation

Transferable targeted attack is more worthy of study because attackers can directly control the unknown model to output the

Table 3 Non-targeted attack success rates (%) on TinyImageNet. The first column introduces the source models and the second row presents the target models. The third row gives the clean accuracy of these target models. Note that the 1,000 testing samples are randomly selected which are all correctly classified by the target models in the non-targeted attack experiments. MI-FGSM, DI-FGSM, and VNI-FGSM are abbreviated as 'MI', 'DI', 'VNI', respectively. '+CKL' represents that our CKL framework is integrated

	Method	Teacher models		Other models			
		ResNet-50	ViT-S	VGG-16	DenseNet-121	ConvMixer	Avg.
Clean accuracy		63.57	39.94	59.32	61.35	57.54	
ResNet-18	MI [14]	56.40	19.00	23.40	23.00	36.80	27.73
	MI+CKL	77.90	80.90	53.60	50.80	68.30	**57.57(+29.84)**
	DI [18]	72.60	24.10	29.80	33.50	45.60	36.30
	DI+CKL	86.00	85.10	63.90	60.90	77.80	**67.53(+31.23)**
	VNI [38]	72.90	23.40	29.10	30.70	44.30	34.70
	VNI+CKL	84.90	85.70	58.90	53.10	74.90	**62.30(+27.60)**
Swin-T	MI [14]	13.70	21.00	10.30	10.00	18.50	12.93
	MI+CKL	43.10	68.50	28.50	28.30	46.30	**34.37(+21.44)**
	DI [18]	21.80	33.10	15.70	15.60	28.50	19.93
	DI+CKL	58.40	77.00	40.50	37.80	59.90	**46.07(+26.14)**
	VNI [38]	13.60	21.70	9.70	9.30	17.30	12.10
	VNI+CKL	42.20	67.00	27.60	25.90	45.00	**32.83(+20.73)**

desired prediction. Specifically, this scenario requires the unknown target model to classify the adversarial examples into a pre-specific class t ($t \neq y$). Therefore, the targeted attack is indeed a more challenging problem.

To evaluate the performance of our method in targeted attack scenario, we employ Logit attack [53] as our baseline method. By following [53], we set the maximum number of iterations to 300, step size to 2/255 and ϵ to 8/255. The target label t ($t \neq y$) is randomly generate for each data pair (x, y). Note that for targeted attack, the attack success rate $tASR$ is computed as

$$tASR = \frac{\#\{x \in \mathcal{X}' | \underset{i}{argmax} F_{tar}(x) = t\}}{\#\{x \in \mathcal{X}'\}}, \tag{9}$$

where $F_{tar}(x)$ denotes the output class of the target model and $\mathcal{X}'$ represents the adversarial examples set. Then, we give the experimental results on CIFAR-10, CIFAR-100 and TinyImageNet dataset.

4.3.1 CIFAR10 results

We utilize the entire CIFAR-10 testing set to generate adversarial examples. The results are shown in Table 4. As can be observed, the tASR scores are usually significantly lower than the corresponding ASR scores, as shown in Table 1, with the same settings. Besides, we can observe that the cross-architecture transfer attack usually leads to lower tASR values, compared to attacking a target model, which is in the same type of DNN architectures. For example, when Logit attack is employed as the attack method and ResNet-18 is utilized as its source model, ViT-S only obtains 9.91% tASR score, because the distinctive features of ResNet-18 and ViT-S tend to be different. On the contrary, when our CKL framework is integrated, the corresponding results obtain large gains, e.g., up to 23.79% improvement for the above example. The last column gives the averaged tASR values of unseen models. Although these test models are not touched in the whole training stage, our method still improves averaged tASRs **at least 20 percentage points**. This phenomenon also indicates that our CKL framework can enable the student (substitute) model to learn common knowledge from multiple teacher models, which significantly improves the adversarial transferability.

4.3.2 CIFAR100 results

CIFAR-100 results are shown in Table 5. The entire CIFAR-100 testing set are utilized to generate adversarial examples. The first column introduces the source models and the second row presents the target models. 'Avg.' represents the averaged tASR values of unseen test models. Targeted attack on CIFAR-100 is a more hard problem, as it contains more categories. From the results, it is observed that the tASR scores on CIFAR-100 are usually lower than the corresponding tASR scores on CIFAR-10. Nevertheless, applying different source models, our method can still outperform the baseline method a lot. Our method can even outperform the baseline method more than 10%, when source/target model is ResNet-18/VGG-16, which validates the effectiveness of our CKL method.

4.3.3 TinyImageNet results

TinyImageNet results are shown in Table 6. Here, we employ the entire validation set, i.e., 10,000 images, to generate adversarial examples to attack the target models. Compared to the attack results on the CIFAR datasets, all the methods possess relatively low tASR scores on TinyImageNet, which implies that pushing a sample to a specific class on TinyImageNet is more difficult. Under such difficult circumstance, our method can still significantly improves the attack performances of the baseline methods.

4.4 Comparison to ensemble attack

A commonly used technique to combine multiple models to generate adversarial examples is ensemble. Our CKL method distills the knowledge of multiple teacher models into one single student model. In this section, we compare our CKL method with ensemble strategy. MI-FGSM is deployed as the attack method. ResNet-50, Inception-v3, Swin-T, and MLPMixer are constructed as the teacher models, which are also the models for ensemble. We conduct the experiment on CIFAR-10 testing set. Ensemble is achieved by utilize the average value of the four networks outputs to generate

Table 4 Targeted attack success rates (%) on CIFAR-10. The first column introduces the source models and the second row presents the target models. Logit attack is selected as the baseline method. '+CKL' represents that our CKL framework is integrated

	Method	Teacher models				Other models				
		ResNet-50	Inception-v3	Swin-T	MLPMixer	VGG-16	DenseNet-121	ConvMixer	ViT-S	Avg.
ResNet-18	Logit	46.33	36.00	15.28	9.46	48.81	58.36	26.58	9.91	35.92
	Logit+CKL	71.01	64.52	52.89	35.73	72.34	77.45	54.42	33.70	**59.48(+23.56)**
Swin-T	Logit	9.28	16.01	99.99$^{\triangle}$	19.57	15.62	10.91	24.16	17.93	17.16
	Logit+CKL	18.62	35.00	93.33	42.86	31.42	23.79	45.86	33.73	**33.70(+15.54)**

Table 5 Targeted attack success rates (%) on CIFAR-100. We generate adversarial examples on the testing set and report the tASR value. '+CKL' represents that our CKL framework is integrated

	Method	Teacher models				Other models				
		ResNet-50	Inception-v3	Swin-T	MLPMixer	VGG-16	DenseNet-121	ConvMixer	ViT-S	Avg.
ResNet-18	Logit [53]	27.19	18.94	3.35	4.50	32.86	31.95	11.54	2.60	19.74
	Logit+CKL	37.56	41.86	9.44	15.17	46.90	41.03	24.11	5.99	**29.51(+9.77)**
Swin-T	Logit [53]	2.45	4.07	96.56$^{\triangle}$	13.13	3.30	2.62	4.42	9.02	4.84
	Logit+CKL	6.75	19.87	12.84	18.08	12.39	8.87	12.48	6.07	**9.95(+5.11)**

Table 6 Targeted attack success rates (%) on TinyImageNet. We generate adversarial examples on the testing set and report the tASR value. '+CKL' represents that our CKL framework is integrated

	Method	Teacher models		Other models			
		ResNet-50	ViT-S	VGG-16	DenseNet-121	ConvMixer	Avg.
ResNet-18	Logit [53]	16.29	1.57	2.49	3.36	4.41	3.42
	Logit+CKL	16.49	16.90	5.42	5.62	8.22	**6.42(+3.00)**
Swin-T	Logit [53]	0.91	1.13	0.61	0.65	1.03	0.76
	Logit+CKL	4.86	9.24	1.88	2.12	3.47	**2.49(+1.73)**

adversarial examples. The results are shown in Table 7. We observe that if the test model is one of the teacher models, ensemble models have better performance. But when tested on unseen models, our CKL method outperforms ensemble strategy, which validates that the effectiveness of the common knowledge learned by our proposed method.

In addition, ensemble strategy has two obvious defects. Firstly, when there exist non-CNN models in ensemble model, it can not employ any intermediate level based attack, because intermediate level based attack needs a pre-specific intermediate layer to get the feature map. But non-CNN models do not possess corresponding feature map. Secondly, directly utilizing ensemble model to generate adversarial examples will cause high computational complexity, especially when the volume of the teacher model is much larger than student model. That is a common circumstance in knowledge distillation. We compare the cost time of generating adversarial examples in the last column, and this experiment is conducted on a single RTX 3080Ti GPU. It is obvious that our method is faster than the ensemble models attack. When using ResNet-18 as the student model, our method (41.10 s) is more than 25× faster than ensemble strategy (1174.65 s).

4.5 Comparison to intermediate feature based attack

To further the broad suitability of our CKL method, we compare it to intermediate feature based attack method. A previously proposed method called intermediate level attack (ILA) [34] finetunes an adversarial example by designing the loss function on the output feature map of an pre-specific intermediate layer of the source model. Recently, ILA-DA [52] employs three novel augmentation techniques to enhance ILA. In this section, we use ILA-DA as the baseline method for comparison. As ILA-DA requires a pre-specific intermediate layer to get the output feature map, it cannot directly employ Transformer-based models as its source model. Therefore, we utilize VGG-16, ResNet-18 and ResNet-50 as the source model to generate adversarial examples. Test model includes CNN and non-CNN models. ASR results are shown in Table 8.

The first column gives the source models and the first row presents the test models. 'Average' represents the averaged ASR values of all test models. From the results, we can see that our CKL method can consistently improve the ILA-DA performance, whatever the test model is. Observing the results of different source models, the averaged ASR value improves **more than 18 percent**, which proves the effectiveness of our CKL method when combining with intermediate level based attack.

4.6 Ablation study

Input gradient distillation scheme Here, the effectiveness of our input gradient distillation scheme is validated. For better comparisons, we firstly introduce several variants of the objective function in learning the input gradients.

(i) The student model is training without gradient learning, i.e., it only employs Eq. (2) as the objective function, which is denoted as 'w/o teacher gradients'.

(ii) The objective function is replaced by the averaged value of multiple teacher models' gradients, i.e., $\|\nabla_x L_S(x) - \bar{g}(x)\|_2^2$, where $\bar{g}(x) = \Sigma_{i=1}^{N} g_i(x)$, which is denoted as 'average of teacher gradients'.

(iii) The objective function is adjusted by gradient projection, which is introduced in Section 3.4.

Table 7 Comparison to ensemble attack on CIFAR-10. We report the averaged attack success rate on the entire testing set. 'CKL' represents that our CKL framework is integrated

		ResNet-50	Inception-v3	Swin-T	MLPMixer	VGG-16	DenseNet-121	ConvMixer	ViT-S	Time/s
Ensemble models		84.14	**97.90**	**100.0**	**99.83**	82.66	74.11	87.82	59.79	1174.65
CKL	ResNet-18	86.30	83.09	82.02	71.78	87.66	**88.37**	81.71	**62.14**	**41.10**
	ResNet-50	**89.11**	85.69	80.12	68.64	86.90	87.97	83.65	58.93	124.30
	VGG-16	79.47	91.89	84.61	68.20	**95.74**	84.07	**91.54**	58.40	54.32

Table 8 Comparison to ILA-DA on CIFAR-10. The first column introduces the source models and the first row presents the test models. We report the averaged attack success rate on the entire testing set. '+CKL' represents that our CKL framework is integrated

	Method	ResNet-34	Inception-v3	MobileNet-v2	DenseNet-121	ConvMixer	ViT-S	Swin-T	Average
ResNet-18	ILA-DA [52]	69.37	59.38	70.58	66.29	55.65	33.42	43.97	56.95
	ILA-DA+CKL	82.74	79.40	85.93	84.38	79.45	59.57	80.00	**78.78(+21.83)**
ResNet-50	ILA-DA [52]	70.96	67.82	78.23	76.98	62.08	35.78	45.21	62.43
	ILA-DA+CKL	82.77	84.97	90.26	86.45	84.30	58.72	81.14	**81.23(+18.80)**
VGG-16	ILA-DA [52]	49.69	89.02	91.66	52.59	80.04	32.34	51.42	62.97
	ILA-DA+CKL	75.80	91.33	94.95	79.81	91.15	55.53	83.33	**81.70(+18.73)**

Here, ResNet-18 is employed as the source model. The teacher models are identical to that in Section 4.1. As can be observed in Table 9, learning the input gradient from teacher models is effective. Besides, our method is more effective than directly learning the average value of teacher gradients.

Effects on hyper-parameter λ. Here, we study the effects of the hyperparameter λ in Eq. (6), which is a factor to balance the two loss terms, i.e., L_{KD} and L_{Grad}. To assess its impacts, we set λ = 1, 5, 10, 50, 100, 500, 1000, 2000. ResNet-18 is employed as the source model and The target models include VGG-16, ResNet-18, ResNet-50, DenseNet-121, Inception-v3, ConvMixer, MLPMixer, Swin-T, and ViT-S. The results are shown in Fig. 4. As can be observed, when λ value is small, the objective function in Eq. (6) is dominanted by the first term, i.e., the objective function of knowledge distillation, and the result remains essentially unchanged. With the increasing of λ, the second term, i.e., the objective function of input gradient distillation, starts to function gradually, and thus the performance gradually increases. However, when λ is relatively large, e.g., $\lambda = 1000$, the attack success rate will decline. Thus, to achieve a good balance between the knowlege distillation and input gradient distillation objectives on all the models, we select $\lambda = 500$ in our experiments.

Selection of the teacher models. To learn a common knowledge from different types of DNN architectures, ResNet-50, Inception-v3, Swin-T, and MLPMixer are employed as the teacher models in our experiments. Here, we study the effects of different selections of the teacher models, i.e.,

(a). 4 CNN models, including ResNet-18, ResNet-50, Inception-v3, and VGG-16;

(b). 2 CNN models and 2 non-CNN models, including ResNet-50, Inception-v3, Swin-T, and MLPMixer;

(c). 4 non-CNN models, including Swin-T, MLPMixer, ConvMixer, and ViT-S.

For convenience, ResNet-18 is employed as the student (source) model. The results are shown in Table 10. As can be observed, when the teacher models are all selected from the non-CNN models, the adversarial transferability to the non-CNN models is relatively high while that to the CNN models is relative low, because the student model learns more bias from the non-CNN models. Besides, we observe an interesting phenomenon that when all the teacher models are selected from CNNs, the attack success rate on the target CNN models are actually not the best. The best results are obtained by selecting 2 CNN and 2 non-CNN models as the teacher models, which implies that learning from both the CNN and non-CNN models is more effective, when attacking the CNN models.

Table 9 Ablation study on different objective functions for input gradient distillation. The source (student) model is ResNet-18. The first row presents the test models

	VGG-16	Swin-T	ViT-S
w/o teacher gradients	86.90	78.85	59.93
Average of teacher gradients	87.65	81.36	61.50
Gradients projection	**87.66**	**82.02**	**62.14**

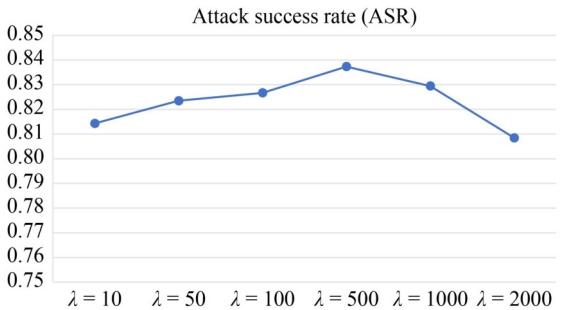


Fig. 4 Attack results with different λ values. The x-axis is the λ value and y-axis denotes the attack success rate. The source (student) model is ResNet-18 and ASR is calculated as the averaged ASR values of nine target models

4.7 Visualization results

We provide several visualization cases to further illustrate the advantage of our method, as shown in Fig. 5. MI-FGSM is employed as the baseline attack method. ResNet-18 is utilized as the substitute model, and ResNet-50 is the test model. '+CKL' denotes that the substitute model is trained by our CKL method. We employ interpretable visualization with GradCAM [54] to obtain a visual explanation for both clean and adversarial images. As can be observed, while both our method and the baseline method could mislead the test model, our method usually suppresses the attention region on the true object, and moves it to other region. This suggests that our method possesses better interpretability and can successfully divert the attention of the network to the wrong region.

5 Discussion and analyses

5.1 Discussion on common knowledge

This paper draws on the idea of common knowledge, to enhance the transferability of adversarial examples. The concept of common knowledge is also used in other tasks, like object caption [55] and etc. This idea is somewhat similar as Multiple Knowledge Representation (MKR) [56], which integrates multiple knowledge representations from different aspects, making the AI techniques more explainable and generalizable. However, our goal is to get better substitute model, which can generate more transferable adversarial examples. Therefore, we design the common knowledge learning framework to distill knowledge from teacher models to the student model, which is used as the substitute model in adversarial attacks. Besides, Since the input gradient is always utilized in typical adversarial attack process, we also design a constraint on the input gradients between the teacher models and the student model, to further promote the transferability of generated adversarial examples.

5.2 Complexity analyses

We propose to learn common knowledge from multiple teacher networks of different architecture. In practice, our

Table 10 Ablation study on the selection of different teacher models. The source (student) model is ResNet-18. The first row presents the test models

	ResNet-18	ResNet-50	DenseNet-121	ConvMixer	Swin-T
Teacher (a)	83.93	79.72	81.87	67.90	51.13
Teacher (b)	**89.51**	**86.30**	**88.37**	81.71	82.02
Teacher (c)	83.69	78.80	82.80	**89.80**	**91.40**

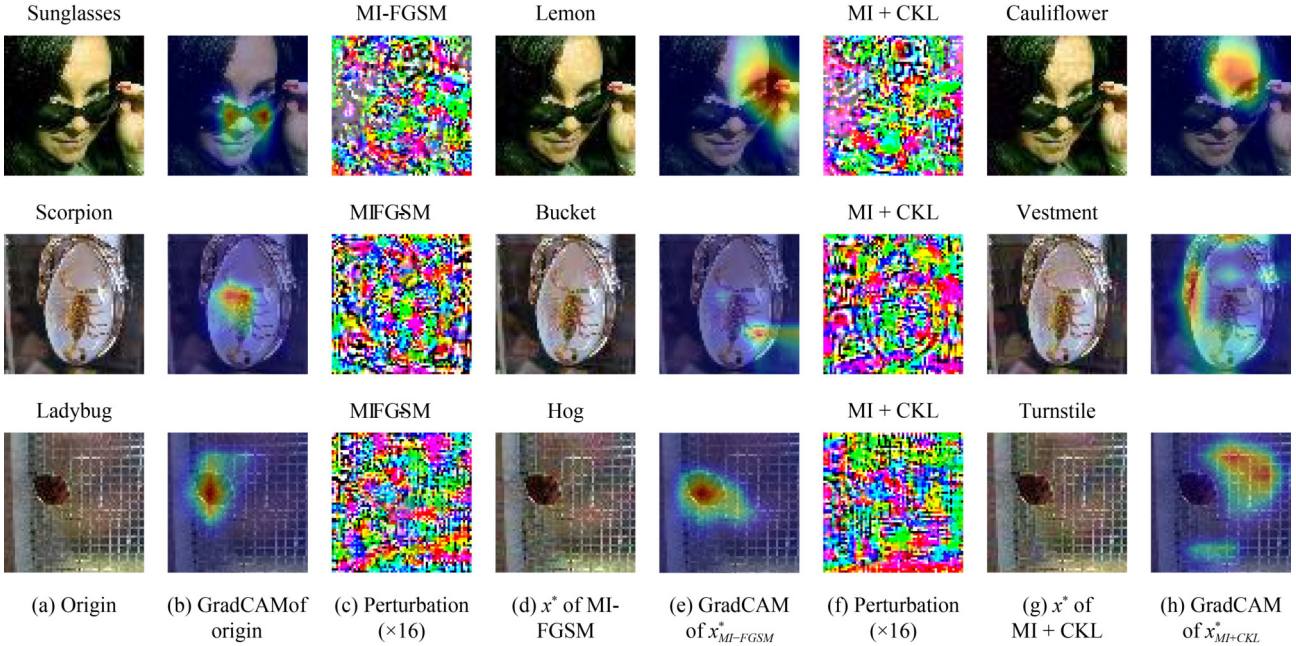


Fig. 5 Visualization results. (a) denotes the clean image x; (b) is the GradCAM of clean image; (c) and (f) are perturbations which are generated by MI-FGSM and MI+CKL method, and the perturbation magnitude is multiplied by 16; (d) and (g) are corresponding adversarial examples; (e) and (h) are GradCAM of the adversarial examples generated by MI-FGSM and MI+CKL, respectively

method requires approximately $N+1$ times of the complexity in the training process, where N is the number of teacher models, compared to the standard training of one DNN model. Fortunately, after the student model is trained, our method can generate adversarial examples with high efficiency as well as a significant improvement compared to the baseline, as demonstrated in the experiments. Since our student model can generate adversarial examples with a much better transferability, we can release the trained student model to the users, such that the users do not need to re-train the student model. Meanwhile, we can utilize N as a hyperparameter to control the tradeoff between generalization ability of the adversarial examples and training complexity, i.e., we can reduce the number of teacher models to reduce the training complexity.

6 Conclusion

In this paper, we analyze the output inconsistency problem in different deep neural networks and propose that the output inconsistency significantly causes the poor transferability of adversarial examples. To alleviate this problem while effectively utilizing the existing DNN models, we propose a common knowledge learning (CKL) framework, which distills the knowledge of multiple teacher models with different architectures into a single student model, to obtain better substitute model for adversarial attack. Specifically, to emphasize the model-agnostic features, the student model is required to learn the outputs from multiple teacher models. To further reduce the output inconsistencies of models and enhance the adversarial transferability, we propose an input gradient distillation to learn the gradients from teacher models. Extensive experiments on CIFAR-10, CIFAR-100, and TinyImageNet have demonstrated the superiority of our proposed work.

Acknowledgments This work was supported in part by the National Natural Science Foundation of China (Grant Nos. 62272020 and U20B2069), in part by the State Key Laboratory of Complex & Critical Software Environment (SKLSDE2023ZX-16), and in part by the Fundamental Research Funds for Central Universities.

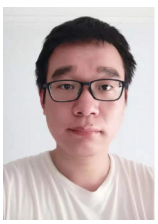
Competing interests The authors declare that they have no competing interests or financial conflicts to disclose.

References

1. Krizhevsky A, Sutskever I, Hinton G E. ImageNet classification with deep convolutional neural networks. In: Proceedings of the 25th International Conference on Neural Information Processing Systems. 2012, 1097–1105
2. Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. In: Proceedings of the 3rd International Conference on Learning Representations. 2015, 1–14
3. He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition. 2016, 770−778
4. Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X, Unterthiner T, Dehghani M, Minderer M, Heigold G, Gelly S, Uszkoreit J, Houlsby N. An image is worth 16x16 words: transformers for image recognition at scale. In: Proceedings of the 9th International Conference on Learning Representations. 2021, 1–21
5. Liu Z, Lin Y, Cao Y, Hu H, Wei Y, Zhang Z, Lin S, Guo B. Swin transformer: hierarchical vision transformer using shifted windows. In: Proceedings of 2021 IEEE/CVF International Conference on Computer Vision. 2021, 9992–10002
6. Sun Y P, Zhang M L. Compositional metric learning for multi-label classification. Frontiers of Computer Science, 2021, 15(5): 155320
7. Ma F, Wu Y, Yu X, Yang Y. Learning with noisy labels via self-reweighting from class centroids. IEEE Transactions on Neural Networks and Learning Systems, 2022, 33(11): 6275−6285
8. Yang Y, Guo J, Li G, Li L, Li W, Yang J. Alignment efficient image-sentence retrieval considering transferable cross-modal representation learning. Frontiers of Computer Science, 2024, 18(1): 181335
9. Hu T, Long C, Xiao C. CRD-CGAN: category-consistent and relativistic constraints for diverse text-to-image generation. Frontiers of

Computer Science, 2024, 18(1): 181304
10. Liang X, Qian Y, Guo Q, Zheng K. A data representation method using distance correlation. Frontiers of Computer Science, 2025, 19(1): 191303
11. Goodfellow I J, Shlens J, Szegedy C. Explaining and harnessing adversarial examples. In: Proceedings of the 3rd International Conference on Learning Representations. 2015, 1−11
12. Lin J, Song C, He K, Wang L, Hopcroft J E. Nesterov accelerated gradient and scale invariance for adversarial attacks. In: Proceedings of the 8th International Conference on Learning Representations. 2020, 1−12
13. Miao H, Ma F, Quan R, Zhan K, Yang Y. Autonomous LLM-enhanced adversarial attack for text-to-motion. 2024, arXiv preprint arXiv: 2408.00352
14. Dong Y, Liao F, Pang T, Su H, Zhu J, Hu X, Li J. Boosting adversarial attacks with momentum. In: Proceedings of 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2018, 9185−9193
15. Yang Y, Huang P, Cao J, Li J, Lin Y, Ma F. A prompt-based approach to adversarial example generation and robustness enhancement. Frontiers of Computer Science, 2024, 18(4): 184318
16. Lu S, Li R, Liu W. FedDAA: a robust federated learning framework to protect privacy and defend against adversarial attack. Frontiers of Computer Science, 2024, 18(2): 182307
17. Zou J, Duan Y, Li B, Zhang W, Pan Y, Pan Z. Making adversarial examples more transferable and indistinguishable. In: Proceedings of the AAAI Conference on Artificial Intelligence. 2022, 3662−3670
18. Xie C, Zhang Z, Zhou Y, Bai S, Wang J, Ren Z, Yuille A L. Improving transferability of adversarial examples with input diversity. In: Proceedings of 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019, 2725−2734
19. Dong Y, Pang T, Su H, Zhu J. Evading defenses to transferable adversarial examples by translation-invariant attacks. In: Proceedings of 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019, 4307−4316
20. Wang X, He X, Wang J, He K. Admix: enhancing the transferability of adversarial attacks. In: Proceedings of 2021 IEEE/CVF International Conference on Computer Vision. 2021, 16138−16147
21. Wu W, Su Y, Lyu M R, King I. Improving the transferability of adversarial samples with adversarial transformations. In: Proceedings of 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021, 9020−9029
22. Mahmood K, Mahmood R, van Dijk M. On the robustness of vision transformers to adversarial examples. In: Proceedings of 2021 IEEE/CVF International Conference on Computer Vision. 2021, 7818−7827
23. Naseer M, Ranasinghe K, Khan S, Khan F S, Porikli F. On improving adversarial transferability of vision transformers. In: Proceedings of the 10th International Conference on Learning Representations. 2022, 1−24
24. Wei Z, Chen J, Goldblum M, Wu Z, Goldstein T, Jiang Y G. Towards transferable adversarial attacks on vision transformers. In: Proceedings of the 36th AAAI Conference on Artificial Intelligence. 2022, 2668−2676
25. Waseda F, Nishikawa S, Le T N, Nguyen H H, Echizen I. Closer look at the transferability of adversarial examples: how they fool different models differently. In: Proceedings of 2023 IEEE/CVF Winter Conference on Applications of Computer Vision. 2023, 1360−1368
26. Yu T, Kumar S, Gupta A, Levine S, Hausman K, Finn C. Gradient surgery for multi-task learning. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. 2020, 489
27. Bruna J, Szegedy C, Sutskever I, Goodfellow I, Zaremba W, Fergus R, Erhan D. Intriguing properties of neural networks. In: Proceedings of the 2nd International Conference on Learning Representations. 2014, 1−10
28. Dong Y, Fu Q A, Yang X, Pang T, Su H, Xiao Z, Zhu J. Benchmarking adversarial robustness on image classification. In: Proceedings of 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020, 318−328
29. Li Y, Li L, Wang L, Zhang T, Gong B. NATTACK: learning the distributions of adversarial examples for an improved black-box attack on deep neural networks. In: Proceedings of the 36th International Conference on Machine Learning. 2019, 3866−3876
30. Cheng M, Le T, Chen P Y, Zhang H, Yi J, Hsieh C J. Query-efficient hard-label black-box attack: an optimization-based approach. In: Proceedings of the 7th International Conference on Learning Representations. 2019, 1−14
31. Shi Y, Han Y, Tian Q. Polishing decision-based adversarial noise with a customized sampling. In: Proceedings of 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020, 1027−1035
32. Zhou L, Cui P, Zhang X, Jiang Y, Yang S. Adversarial Eigen attack on BlackBox models. In: Proceedings of 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022, 15233−15241
33. Lord N A, Mueller R, Bertinetto L. Attacking deep networks with surrogate-based adversarial black-box methods is easy. In: Proceedings of the 10th International Conference on Learning Representations. 2022, 1−17
34. Huang Q, Katsman I, Gu Z, He H, Belongie S J, Lim S N. Enhancing adversarial example transferability with an intermediate level attack. In: Proceedings of 2019 IEEE/CVF International Conference on Computer Vision. 2019, 4732−4741
35. Yang D, Li W, Ni R, Zhao Y. Enhancing adversarial examples transferability via ensemble feature manifolds. In: Proceedings of the 1st International Workshop on Adversarial Learning for Multimedia. 2021, 49−54
36. Gumus F, Amasyali M F. Exploiting natural language services: a polarity based black-box attack. Frontiers of Computer Science, 2022, 16(5): 165325
37. Yang B, Zhang H, Li Z, Zhang Y, Xu K, Wang J. Adversarial example generation with adabelief optimizer and crop invariance. Applied Intelligence, 2023, 53(2): 2332−2347
38. Wang X, He K. Enhancing the transferability of adversarial attacks through variance tuning. In: Proceedings of 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021, 1924−1933
39. Yuan H, Chu Q, Zhu F, Zhao R, Liu B, Yu N. AutoMA: towards automatic model augmentation for transferable adversarial attacks. IEEE Transactions on Multimedia, 2023, 25: 203−213
40. Tolstikhin I O, Houlsby N, Kolesnikov A, Beyer L, Zhai X, Unterthiner T, Yung J, Steiner A, Keysers D, Uszkoreit J, Lucic M, Dosovitskiy A. MLP-mixer: an all-MLP architecture for vision. In: Proceedings of the 35th International Conference on Neural Information Processing Systems. 2021, 24261−24272
41. Gou J, Yu B, Maybank S J, Tao D. Knowledge distillation: a survey. International Journal of Computer Vision, 2021, 129(6): 1789−1819
42. Passban P, Wu Y, Rezagholizadeh M, Liu Q. ALP-KD: attention-based layer projection for knowledge distillation. In: Proceedings of the 35th AAAI Conference on Artificial Intelligence. 2021, 13657−13665
43. Chen D, Mei J P, Zhang Y, Wang C, Wang Z, Feng Y, Chen C. Cross-layer distillation with semantic calibration. In: Proceedings of the 35th AAAI Conference on Artificial Intelligence. 2021, 7028−7036
44. Lee S, Song B C. Graph-based knowledge distillation by multi-head attention network. In: Proceedings of the 30th British Machine Vision Conference. 2019, 141
45. Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network. 2015, arXiv preprint arXiv: 1503.02531
46. Liu B, Liu X, Jin X, Stone P, Liu Q. Conflict-averse gradient descent for multi-task learning. In: Proceedings of the 35th International Conference on Neural Information Processing Systems. 2021, 1443
47. Krizhevsky A. Learning multiple layers of features from tiny images. University of Toronto, Dissertation, 2009
48. Huang G, Liu Z, Van Der Maaten L, Weinberger K Q. Densely connected convolutional networks. In: Proceedings of 2017 IEEE Conference on Computer Vision and Pattern Recognition. 2017,

2261−2269
49. Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z. Rethinking the inception architecture for computer vision. In: Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition. 2016, 2818−2826
50. Sandler M, Howard A G, Zhu M, Zhmoginov A, Chen L C. MobileNetV2: inverted residuals and linear bottlenecks. In: Proceedings of 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2018, 4510−4520
51. Ng D, Chen Y, Tian B, Fu Q, Chng E S. Convmixer: feature interactive convolution with curriculum learning for small footprint and noisy far-field keyword spotting. In: Proceedings of the ICASSP 2022−2022 IEEE International Conference on Acoustics, Speech and Signal Processing. 2022, 3603−3607
52. Yan C W, Cheung T H, Yeung D Y. ILA-DA: improving transferability of intermediate level attack with data augmentation. In: Proceedings of the 11th International Conference on Learning Representations. 2023, 1−25
53. Zhao Z, Liu Z, Larson M A. On success and simplicity: a second look at transferable targeted attacks. In: Proceedings of the 35th International Conference on Neural Information Processing Systems. 2021, 468
54. Selvaraju R R, Cogswell M, Das A, Vedantam R, Parikh D, Batra D. Grad-CAM: visual explanations from deep networks via gradient-based localization. International Journal of Computer Vision, 2020, 128(2): 336−359
55. Wu Y, Jiang L, Yang Y. Switchable novel object captioner. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2023, 45(1): 1162−1173
56. Yang Y, Zhuang Y, Pan Y. Multiple knowledge representation for big data artificial intelligence: framework, applications, and case studies. Frontiers of Information Technology & Electronic Engineering, 2021, 22(12): 1551−1558



Ruijie YANG received the BS degree from Beihang University, China in 2018. He is now a PhD student at the Laboratory of Intelligent Recognition and Image Processing, Beihang University, China. His research interests are adversarial machine learning, interpretability of DNN and model security.

Yuanfang GUO received his BEng degree in computer engineering and PhD degree in electronic and computer engineering from The Hong Kong University of Science and Technology, China in 2009 and 2015, respectively. He is currently an associate professor with the School of Computer Science and Engineering, Beihang University, China. His current research interests include multimedia security, artificial intelligence security, graph neural networks. He has published over 90 scientific papers.

Junfu WANG received the BE degree in software engineering from Chongqing University, China in 2019. He is currently working toward the PhD degree with the Laboratory of Intelligent Recognition and Image Processing, School of Computer Science and Engineering, Beihang University, China. His research interests include graph representation learning, in particular, on large-scale network, heterogeneous network, network with heterophily.

Jiantao ZHOU received his BEng degree from the Department of Electronic Engineering, Dalian University of Technology, China in 2002, the MPhil degree from the Department of Radio Engineering, Southeast University, China in 2005, and the PhD degree from the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, China in 2009. He is currently a professor with the Department of Computer and Information Science, Faculty of Science and Technology, University of Macau, China. His research interests include multimedia security and forensics, multimedia signal processing, artificial intelligence, and big data.

Yunhong WANG (Fellow, IEEE) is currently a professor at Beihang University, China, where she is also the Director of the Laboratory of Intelligent Recognition and Image Processing, School of Computer Science and Engineering. She received the PhD degree in electronic engineering from the Nanjing University of Science and Technology, China in 1998. She was with the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, China from 1998 to 2004. Her research interests include biometrics, pattern recognition, computer vision, data fusion, and image processing. She has published more than 300 research papers in international journals and conferences. She has served on the editorial board of IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Biometrics, Behavior, and Identity Sciences, and Pattern Recognition. She is a Fellow of IEEE, IAPR, CAAI, and CCF.