Expert Systems With Applications 265 (2025) 126025

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Reusable generator data-free knowledge distillation with hard loss simulation for image classification ☆

Yafeng Sun [a], Xingwang Wang [a,b,*], Junhong Huang [a], Shilin Chen [a], Minghui Hou [a]

[a] College of Computer Science and Technology, Jilin University, Changchun 130012, China
[b] Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, Changchun 130012, China

ARTICLE INFO

Keywords:
Data-free
Knowledge distillation
Generator
Adversarial example attack
Image classification

ABSTRACT

In many image classification scenarios where knowledge distillation (KD) is applied, multiple users need to train various student models that conform to the device's computational limitations at different times. However, due to security concerns, existing data-free KD (DFKD) methods discard the trained generator after completing a task, preventing its reuse in training subsequent students and resulting in wasted resources. Furthermore, the unlabeled nature of fake images eliminates the hard loss that plays a crucial role in data-driven KD. This paper introduces the reusable generator DFKD with hard loss simulation (RG-HLS) to address these challenges. RG-HLS employs a generator consisting of two generative networks and a decoder. The decoder and one of the generative networks can be reused in subsequent tasks. The adversarial example attack is also introduced to mimic the behavior of the teacher model in predicting real images, ensuring that DFKD exhibits a hard loss similar to that of data-driven KD. ,Extensive experiments across CIFAR-10, CIFAR 100, MNIST, SVHN, and Tiny-ImageNet datasets validate the proposed method's superior performance. The reusable generator accelerates convergence without compromising security, and integrating the hard loss further boosts accuracy. Code is available at https://github.com/sunyafeng-jlu/RG-HLS.

1. Introduction

The rise of artificial intelligence and its widespread application has been accompanied by remarkable achievements in deep learning (DL). However, with increasing model sizes, deploying DL models on resource-limited embedded or edge devices has become challenging (Wang, Sun, Chen and Xu, 2024). To address this issue, knowledge distillation (KD) has emerged as a promising technique for model compression. KD aims to transfer knowledge from large-scale models to lightweight ones in a data-driven manner, enabling the deployment of lightweight models (Gou, Hu, Sun, Wang and Ma, 2024). Currently, data-driven KD has been successfully applied in various fields, including image classification (Ma, Zhang, Cao, Li, & Gao, 2024), traffic prediction (Li, Li, Yan, Liu, & Liu, 2024), and fault diagnosis (Guo, Li and Shen, 2024).

In recent years, privacy and security issues in the image classification domain have garnered significant attention (Xie, Chen, Wu, & Li, 2024; Zhang, Wu, Tian, Zhang, & Lu, 2021). Many application scenarios involve sensitive images that cannot be disclosed, such as medical images, face images, etc. In such cases, data-driven KD cannot cope with the absence of available data (Wang, Qian, Liu, Rui and Wang, 2024). Fortunately, pre-trained models on these images do not infringe on user privacy and can be published for downstream tasks. Data-free knowledge distillation (DFKD) is developed to address this scenario. Typically, DFKD utilizes a generator to synthesize fake images, which are then used to drive the migration of knowledge (Chen et al., 2024).

The method for synthesizing fake images determines the DFKD performance ceiling (Li, Zhou et al., 2023). Current methods either start with randomly generated images and refine them Yin et al. (2020) or use generative networks for images generation (Chen et al., 2024; Shao, Zhang and Wang, 2023). The former method generates numerous random images, potentially slowing the distillation pipeline (Yu, Chen, Han, & Jiang, 2023). The latter methods often rely on generative adversarial networks (GANs) (Yilmaz & Korn, 2024), which have a generative network creating fake images to fool the discriminator and a discriminator distinguishing real from fake images (Yang, Tang, Dang, Chen, & Chambers, 2024). The key difference between DFKD

☆ This research was supported in part by the National Natural Science Foundation of China (NSFC) (Grants No. U19A2061, No. 61902143), in part by National Key Research and Development Program of China (No. 2023YFB4502304).

* Corresponding author at: College of Computer Science and Technology, Jilin University, Changchun 130012, China.
E-mail addresses: yfsun22@mails.jlu.edu.cn (Y. Sun), xww@jlu.edu.cn (X. Wang), huangjh23@mails.jlu.edu.cn (J. Huang), chensl22@mails.jlu.edu.cn (S. Chen), houmh21@mails.jlu.edu.cn (M. Hou).

https://doi.org/10.1016/j.eswa.2024.126025
Received 17 June 2024; Received in revised form 13 November 2024; Accepted 1 December 2024
Available online 7 December 2024

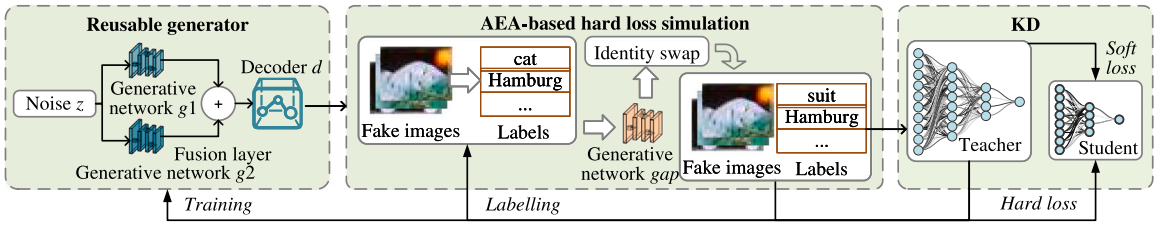


Fig. 1. Overview of the proposed RG-HLS.

and GAN is that the teacher model in DFKD assumes the role of the discriminator, which is not trained (Chen et al., 2019). Generator-based DFKD methods can produce a substantial number of fake images simply by training a competent generative network, making them significantly more efficient than DFKDs optimized for random images (Yu et al., 2023).

In real-world KD scenarios, deployment engineers across different organizations often need to train multiple student models at various times for devices with varying computational capabilities based on a published teacher model, without knowing which student model to train next. As a result, a generative network that is well trained in one KD task can be useful in other tasks as well. However, current Generator-based DFKDs discard the trained generative network after completing a KD task and do not reuse it for subsequent tasks (Chen et al., 2019, 2024; Shao, Zhang, Wang, 2023; Yu et al., 2023), leading to a waste of resources. This is due to deployment specificity and security considerations, as saving trained generative networks carries the risk of privacy leakage. Additionally, in data-driven KD methods, the hard loss obtained from ground truth labels encourages the student model to achieve better generalization and improve its accuracy. Yet most DFKDs generate unlabeled fake images unsupervisedly (Chen et al., 2019, 2024; Yin et al., 2020; Yu et al., 2023), preventing hard loss computation. Even supervised DFKDs (Luo, Sandler, Lin, Zhmoginov, & Howard, 2020; Shao, Zhang, Wang, 2023), which generate labeled images but cannot match ground truth labels' effect. We aim to address these two challenges. The most intuitive solution is to (1) reutilize the pre-trained generative network in a new DFKD task and (2) employ the teacher model to label the fake images. Nevertheless, these simple approaches are not feasible:

- Reusing or publishing pre-trained generative networks may violate the privacy requirements of data-free scenarios. If the pre-trained generative networks are exposed, it is feasible for attackers to utilize them to generate images, which can then be employed in data-driven AI attacks (Mukherjee, 2023; Zhao et al., 2023). Our empirical study in Section 5.3 demonstrates that compared to random images, the fake ones cause about a 20% decrease in image classification accuracy.
- The labels labeled by the teacher model in DFKD are soft labels. Unlike data-driven KD settings where the teacher model predicts whether images are correct or incorrect, these labels ensure that the teacher's predictions are always accurate in DFKD. Consequently, the student model is unable to learn from these images, let alone improve its accuracy.

To address the first challenge, we drew inspiration from image fusion techniques (Guo, Zhan et al., 2024) and made modifications to the generator's structure. The improved generator consists of two generative networks, a feature fusion strategy, and a decoder, As demonstrated by the reusable generator in Fig. 1. A portion of its architecture draws on DenseFuse (Li & Wu, 2018), which focuses on fusing infrared and visible images by encoding and fusing the features of multiple images. In our method, we replace the encoder with two generative networks for reusability. These networks generate images treated as encoded features, fused to form a unified feature, which is decoded to produce fake images. Once the current DFKD task is complete, the decoder and one of the generative networks, which is more suitable for security, can be repurposed for another task.

To address the second challenge, An in-depth analysis of the teacher's prediction behavior in data-driven KD is conducted. We observed the teacher's high confidence even when making incorrect prediction. This finding aligns with the objective of the adversarial example attack (AEA) approach, which introduces small perturbations to input images. These perturbations often manifest as noise in computer vision, causing the model to confidently output incorrect classification results. The perturbed images remain easily distinguishable to humans (Macas, Wu, & Fuertes, 2024). Therefore, we integrate AEA into DFKD to mimic the behavior of the teacher model in data-driven KD when making incorrect predictions. This enables DFKD to obtain the ground truth label and hard loss similar to that of data-driven KD.

The primary contributions of this paper are as follows:

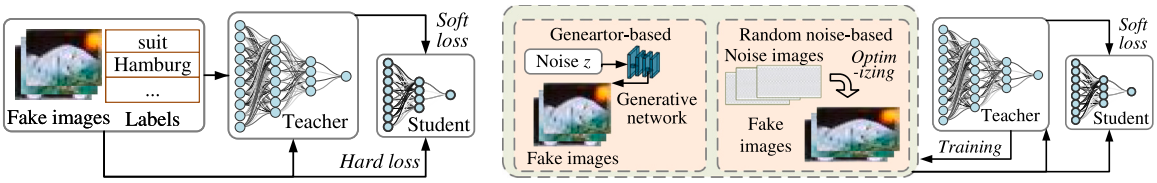
- Reusable Generator. The novel generator architecture described earlier allows the generator to be reused. This architecture eliminates the need to discard the entire trained generator, thereby reducing waste.[1] Not only does it enhance the classification accuracy and convergence speed of the generator, but it also satisfies security requirements.
- AEA-based hard loss simulation. This hard loss simulation using AEA provides DFKD with hard labels akin to data-driven KD, enabling the computation of hard loss. It elevates the accuracy of DFKD compared to versions that rely solely on soft loss.
- Extensive experiments. Leveraging the aforementioned innovations, we present the reusable generator data-free knowledge distillation with hard loss simulation (RG-HLS). We conduct comprehensive experiments on CIFAR-10, CIFAR-100, MNIST, SVHN, and Tiny-ImageNet image classification datasets, demonstrating the competitiveness of RG-HLS.

The remainder of this paper is organized as follows: Section 2 provides a brief overview of KD and DFKD. Section 3 describes the related work. Section 4 delves into the proposed RG-HLS. Section 5 presents comparisons with state-of-the-art (SOTA) DFKDs and experimental evaluations on the effectiveness of the strategy. Section 6 wraps up this paper with a summary.

2. Background

This section describes the basic concepts of KD and DFKD in the field of image classification, which are schematically shown in Fig. 2.

[1] Here, the "reducing waste" pertains to the fact that a trained generator is partially reused for other DFKD tasks, thereby avoiding completely discarding it entirely, rather than diminishing the computational resources allocated for its training.



(a) KD (b) DFKD

Fig. 2. A brief illustration of KD and DFKD.

2.1. Knowledge distillation

In KD (Li, Li et al., 2023), a sizeable and excessively parameterized teacher model serves to extract a diverse array of knowledge, subsequently utilizing this knowledge to train a lightweight student model. Notably, the teacher model remains static throughout this entire process, as illustrated in Fig. 2(a). Without loss of generality, KD trains student by minimizing the disparities between student s and teacher t outputs and compelling student to learn the ground truth labels y :

$$l_s = \alpha \cdot \underbrace{\tau^2 \cdot \mathrm{KL}(\mathrm{softmax}(O_s/\tau), \mathrm{softmax}(O_t/\tau))}_{\text{soft loss (KD loss)}} + \beta \cdot \underbrace{\mathrm{CrossEntropy}(\mathrm{softmax}(O_s, y))}_{\text{hard loss}}, \tag{1}$$

where O_s and O_t are logits of student and teacher respectively, τ, α, β are hyper-parameters, and CrossEntropy$(\cdot)$, KL$(\cdot)$ are cross entropy loss function and KL-divergence loss function respectively. The part controlled by α is known as soft loss or KD loss, which actually makes the output of the student to be closer to the output of the teacher to learn the knowledge. The part controlled by β is called hard loss, which drives the student to learn from truth labels.

2.2. Data-free knowledge distillation

Unlike KD, DFKD does not have real images available. Instead, it must synthesize fake images to support the distillation process, as illustrated in Fig. 2(b). As mentioned in Section 1, DFKDs can be classified into two categories: those that begin with random images and those that utilize a generator to create fake images (Shao, Zhang, Yin and Wang, 2023). The latter approach holds more promise, as it generates large amounts of images $\widetilde{x}$ in a shorter period of time:

$$\widetilde{x} = g(z), \tag{2}$$

where z is the random noise and g denotes the generator. The fake images $\widetilde{x}$ are inducted by the teacher and student to obtain the logit output $\widetilde{O_t}$ and $\widetilde{O_s}$:

$$\begin{cases} \widetilde{O_t} = t(\widetilde{x}) \\ \widetilde{O_s} = s(\widetilde{x}) \end{cases}. \tag{3}$$

Generally g generates fake images without labels, so the second half of Eq. (1) is invalid. $\widetilde{O_t}$ and $\widetilde{O_s}$ are involved in the first half of Eq. (1) in order to instruct the student to learn about the teacher:

$$\widetilde{l_s} = \tau^2 \cdot \mathrm{KL}(\mathrm{softmax}(\widetilde{O_s}/\tau), \mathrm{softmax}(\widetilde{O_t}/\tau)). \tag{4}$$

3. Related work

Since its inception, KD has undergone extensive enhancements to boost its performance. These advancements encompass the incorporation of intermediate features (Gou, Sun, Yu, Wan, & Tao, 2023), image generation techniques (Zhao, Sun, Dong, Yu and Wang, 2022), and various other innovations. The latest iterations of KD focus on smarter learning methodologies. For instance, DKD leverages logit-level information to guide students in learning from the instructor, thereby enhancing image classification accuracy without introducing additional overheads (Zhao, Cui, Song, Qiu and Liang, 2022). Furthermore, FFKD allows instructors to receive feedback from students, enabling adaptive adjustments to the learning mode (Gou, Chen et al., 2024).

While these improved approaches have delivered commendable results in image classification tasks, they are not applicable in scenarios where, due to privacy concerns, the KD process excludes real images. The development of DFKD is specifically aimed at allowing the KD paradigm to function effectively in data-free scenarios. DAFL (Chen et al., 2019) is the pioneering effort in generator-based DFKDs, introducing two regularization losses for activation and prediction to assist in generating fake images. DeepInv (Yin et al., 2020) ensured that the fake images exhibited the same mean and variance as the real images when passing through the BN layer of the teacher, making the fake images more aligned with the real images. It is worth noting that DeepInv (Yin et al., 2020) started the DFKD approach with random images, but its concepts have been widely adopted in generator-based DFKDs. CMI (Fang et al., 2021) achieved satisfactory results in mitigating the schema-collapse problem by modeling data diversity as an optimizable objective. The BNS, class prior, and KL adversarial distillation losses were integral to the optimization of CMI (Fang et al., 2021), enhancing the stability of DFKD. Guo et al. contend that the fake samples produced by the generator exhibit a non-smooth distribution, which undermines the accuracy of the student (Patel, Mopuri, & Qiu, 2023). In response, they introduce a meta-learning framework, where knowledge acquisition and retention are framed as meta-training and meta-testing, respectively. The MAD approach maintains exponential moving averages of the generator to synthesize samples and train the student models (Do et al., 2022).

These existing DFKD methods strive to enhance the classification accuracy of student models in data-free contexts from diverse angles. Our objective aligns with theirs, albeit with a distinct focus: while MLISU emphasizes the detrimental impact of unstable pseudo-sample distributions and MAD highlights distribution skewness, we posit that the absence of hard losses represents a crucial barrier to improving the classification accuracy of the student model. As illustrated in Table 1,[2] when KDs that are not considered for data-free scenarios lack hard losses, their classification accuracies are significantly degraded, whereas, for DFKDs, the synthesized samples lack real labels, necessitating the abandonment of hard losses, which subsequently results in a crucial learning source being forfeited compared to data-driven KDs. Moreover, we observe that existing methods are compelled to discard the trained generator due to privacy limitations. However, if this generator could be reused for the subsequent classification task, it would not only be put to good use but also expedite convergence.

In summary, our motivation, like those of related works, is to improve the final classification accuracy of the student model in DFKD, with the difference that we approach it from the perspective of the hard loss and reuse of the generator. To the best of our knowledge, this paper is the first work that considers those two perspectives.

[2] All methods and KD tasks are executed on the CIFAR-100 dataset. OFD (Heo et al., 2019) utilizes intermediate information and relies heavily on hard loss, the absence of which will result in student accuracy approaching random guessing.

Table 1
Accuracy comparison of the advanced KD methods without modification and without hard loss.

Teacher	Student		Image classification accuracy		
			KD	DKD	OFD
ResNet32 × 4	ResNet8 × 4	Without modification	73.33	76.32	74.95
		Without hard loss	73.27	75.26	1.48
WRN40-2	WRN16-2	Without modification	74.92	76.24	75.24
		without hard loss	74.22	75.08	1.76

4. RG-HLS

In this section, we elaborate on the proposed RG-HLS, covering the reusable generator, AEA-based hard loss simulation, and other specific details.

4.1. Overview

To address the challenges of non-reusability of the generator due to privacy constraints and the absence of hard loss caused by unlabeled fake images, we present a modified generator structure and employs AEA to mimic the behavior of the teacher for predicting real images. The framework of our proposed method is depicted in Fig. 1.

Similar to traditional generator-based DFKDs, the generator in RG-HLS accepts noise as input and produces fake images as output. It comprises two generative networks (g_1, g_2), a fusion layer, and a decoder (d). Specifically, g_1, g_2, and d are randomly initialized and trained for the first task of RG-HLS. For subsequent tasks, a reusable generative network (g_1 or g_2) and d are pre-trained. The rationale behind this design, and which generative network to publish in the first task—is explained in Section 4.2, while the security of the reusable generator is validated in Section 5.3.

After acquiring fake images, the teacher makes labels for them. The fake images along with their corresponding labels are then processed through the generative adversarial perturbation model gap, which is used for AEA and outputs the attacked images. Finally, the identity swap operation is conducted to yield the images and labels that are ultimately utilized to steer the KD. This is the proposed AEA-based hard loss simulation, which aims to simulate hard labels using teacher and AEA, enabling the subsequent KD process to incorporate the hard loss. This strategy will be further detailed in Section 4.3.

Additionally, RG-HLS operates in two modes: Anew and Reuse. In Anew mode, g_1, g_2, and d are all trained from scratch, and after the task, d and the reusable generative network are released. In Reuse mode, g_1 and d are reused from the Anew mode. It is important to note that the classification accuracy in Reuse mode serves as the accuracy indicator of RG-HLS performance.

4.2. Reusable generator

Generator-based DFKDs primarily consist of a generative network that receives noise and produces fake images, as specified in Eq. (2). Generally, as g is progressively optimized, it can deceive the teacher, even if $\widetilde{x}$ is not increasingly converging to real images x. This is because optimizing g involves acquiring information about teacher's inference of $\widetilde{x}$. Hence, if the trained g is made public, the images it generates can aid data-driven attack methods against DL models. The most conservative estimate is that attacking the teacher in the DFKD task of the trained g is effective, and the generalization of g can potentially impact other models. This is why numerous generative-based DFKDs do not consider reusing generators. We aim to not only reuse generators but also ensure security. Therefore, reusing parts of generators appears to be a promising approach.

Inspired by DenseFuse (Li & Wu, 2018), a pioneering work in image fusion (Tang, Xiang, Zhang, Gong, & Ma, 2023; Zhou, Li, Lu, Cheng, & Zhang, 2023), we have made improvements to the generator's structure to easy reusability. The model architecture of DenseFuse consists of an encoder, fusion layer, and decoder. Multiple images are fed into DenseFuse (Li & Wu, 2018), and after features are extracted by the encoder, the fusion layer combines these features. Finally, the decoder processes the features and generates the output images. In this paper, we extend the generator to include two generative networks, g_1, and g_2, which produce two images from the same noise input. These two images are treated as features, which are combined into a single feature after passing through the fusion layer. For simplicity, the fusion layer in this paper only sums up the two feature maps. The resulting features are then processed by decoder d to generate the final fake images. In accordance with the flowchart in Fig. 1, the enhanced generator generates the fake image as:

$$\widetilde{x} = d(g_1(z) + g_2(z)) \tag{5}$$

The structure of the reusable generator is illustrated in Fig. 3(a). The generative networks and decoder each consist of four convolutional layers. The fusion layer $\oplus$ combines the features from the two generative networks through summation. In Fig. 3(a), B, H, and W represent the batch size, the height of the image, and the width of the image, respectively. The decoder d is primarily comprised of four convolutional modules. The first three convolutional layers are succeeded by a BN layer and a ReLU activation layer, whereas the final convolutional layer is followed by a Sigmoid activation function. This choice is necessitated by the fact that the augmented real image data's values lie between 0 and 1, rendering the ReLU function unsuitable. The decoder receives features with dimensions B x 3 x H x W and produces a fake image of identical dimensions, thereby ensuring no disruption to the knowledge distillation process at the data format level. It is worth noting that, in contrast to DenseFuse, our decoder is designed to be more streamlined, as it collaborates with the preceding generator to generate the fake image, rather than being tasked with the actual image fusion process. Fig. 3(b) displays the architectures of g_1, g_2, both comprising a linear layer and three convolutional layers. The linear layer functions to amplify the noise, whereas the subsequent three convolutional layers produce fake images with dimensions B x 3 x H x W. It is important to note that the generator structure depicted in Fig. 3(b) is not a novel contribution of this paper but has been selected for comparative fairness; this structure is widely employed by most DFKDs for image classification, including the methods compared in this study.

The reusable generator comprises g_1, g_2, and d, and we enable d and one of the generative networks to be utilized by subsequent tasks. Consequently, during the first task, only one generative network of the trained reusable generator is discarded, while the remaining components can be published. In the absence of one generative network, an attacker cannot exploit the reusable generator to produce reasonable images, thereby safeguarding data security. Specifically, at the commencement of RG-HLS, the DFKD user must select either the Anew mode or the Reuse mode:

- In Anew mode, g_1, g_2, and d undergo random initialization and undergo training with the knowledge distillation process. Upon completing the student model's training, one of $[g_1, g_2]$ and d are set aside for future tasks. The Anew mode is ideal for users when they first train the student model.
- In Reuse mode, g_1 is randomly initialized, whereas g_2 and d are sourced from the published components in the task executed in Anew mode. The rest of the process mirrors that of Anew mode.

To exemplify the application contexts of these two modes, consider a scenario where a user intends to distill ResNet18 using a ResNet34 model pre-trained on CIFAR-100, and this is their inaugural DFKD task. In such a case, they must proceed with distillation in Anew mode. Upon completion, not only is ResNet18 trained, but the reusable generator is also refined, with one generative network and d being

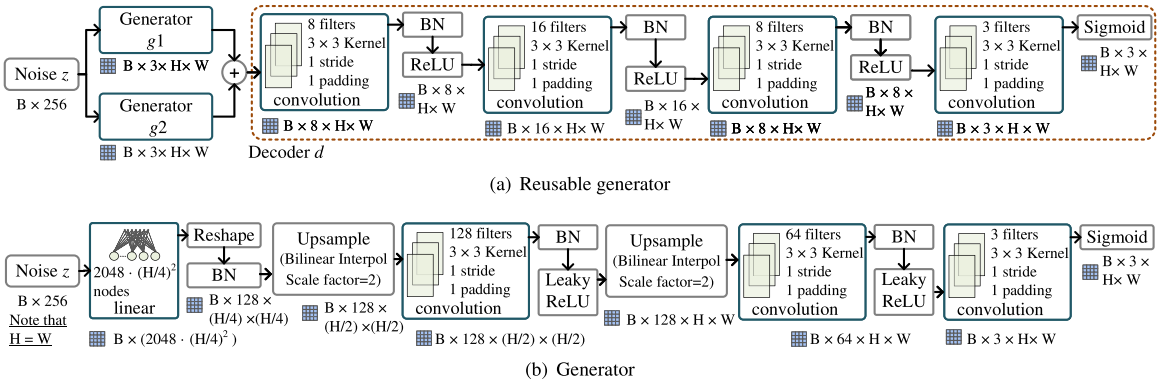


Fig. 3. Structure of the proposed reusable generator.

released. If the user subsequently needs to distill WRN40-1 using a WRN40-2 model pre-trained on CIFAR-100, they can opt for Reuse mode, leveraging the generative network and d released from the ResNet34-ResNet18-CIFAR-100 task.

Next, we need to determine which generative network the Anew model will output. An important consideration is that feature maps tend to produce higher activation values when the input images are genuine rather than random vectors (Chen et al., 2019). Therefore, we generate features using both g_1 and g_2, and each feature is then combined with a random vector. After processing by the decoder, we obtain two batches of images: b_1 and b_2. These images will be evaluated by the teacher, and its feature map serves as a metric for determining which generative network to publish:

$$\begin{cases} b_1 = d(g_1(z) + r) \\ b_2 = d(g_2(z) + r) \end{cases}, \tag{6}$$

$$rg = \begin{cases} g_1, & \text{if } \| f_t^{b_1} \|_1 < \| f_t^{b_2} \|_1 \\ g_2, & \text{otherwise} \end{cases}, \tag{7}$$

where rg is the reusable, publishable generative network that can be reused in other tasks. r is a random vector with the same dimensions as the fake image. $f_t^{b_1}$ and $f_t^{b_2}$ represent the feature maps prior to the linear layer when the teacher reasons about b_1 and b_2, respectively. $\|\cdot\|_1$ denotes the L1 norm. In this paper, three loss functions are employed to optimize the generator:

$$\begin{cases} l_{oh}(\widetilde{x}) = \text{CrossEntropy}(\text{softmax}(\widetilde{O_t}, \widetilde{y})) \\ l_{adv}(\widetilde{x}) = -\text{JS}(\widetilde{O_s}/\tau, \widetilde{O_t}/\tau) \\ l_{bn}(\widetilde{x}) = \sum_l [\|(1-m)\cdot\mu_a^l + m\cdot\mu_{ft}^l - \mu_{bn}^l\|_1 + \| (1-m)\cdot\sigma_a^l + m\cdot\sigma_{ft}^l - \sigma_{bn}^l \|_1] \end{cases}, \tag{8}$$

where JS(·) represents the Jensen–Shannon divergence and $\widetilde{y}$ represents the label assigned by t to $\widetilde{x}$. μ_{bn}^l and σ_{bn}^l are the mean and variance, respectively, of the lth BN layer in t. μ_{ft}^l and σ_{ft}^l are the mean and variance of $\widetilde{x}$ after passing through the lth BN layer of t. It is important to note that the noise z must also be optimized. Furthermore, the optimization of noise z is essential. The optimization method for the generator is not the main focus of this paper, readers are referred to FM (Fang et al., 2022) as our optimization approach aligns with it.

4.3. AEA-based hard loss simulation

Table 1 highlights the importance of hard loss in data-driven KD. However, DFKD is unable to utilize it due to the absence of labels. Our objective is to assign labels to the fake images by mimicking the teacher's reasoning behavior in data-driven KD. Fig. 4[3] reveals that the teacher's behavior can be categorized into two scenarios: correct prediction and incorrect prediction.

As Fig. 4(a) illustrates, when the teacher makes a correct prediction, it tend to exhibit high confidence, with a confidence level exceeding 0.9. Conversely, when the teacher makes an incorrect prediction, their confidence level is lower than that of a correct prediction, but it remains relatively high (as shown in Fig. 4(b)). In other words, the teacher confidently provides an incorrect prediction result. Additionally, we conducted experiments using various architectures such as VGG, MobileNet, and others on the CIFAR-10 and CIFAR-100 datasets, beyond those depicted in Fig. 4(a). We observed 4 key phenomena:

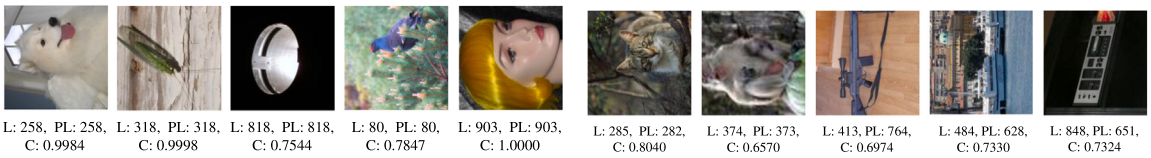
(1) The confidence is extremely high when the prediction is accurate.
(2) Even when the prediction is incorrect, it still maintains a high confidence level.
(3) The confidence level is generally higher when the prediction is correct compared to when it is incorrect.
(4) When predicting the train dataset, the error rate of the pre-trained model is impressively low.[4]

The objective is to create labels for the fake images that closely resemble the four mentioned phenomena. If only teacher is used to label the fake images, it can only mimic phenomenon (1). At this point, there can be no prediction error. Fortunately, phenomenon (2) is exactly what the adversarial example attack (Han, Lin, Shen, Wang, & Guan, 2023) is trying to accomplish, which involves adding a subtle perturbation to the image. This perturbation causes the model to output incorrect predictions, but it does not affect human manual recognition. Fig. 5 illustrates the prediction results for images after AEA.

Fig. 5(b) illustrates the impact of five AEA methods, where GAP (Poursaeed, Katsman, Gao, & Belongie, 2018) and DeepFool (Moosavi-Dezfooli, Fawzi, & Frossard, 2016) offer flexibility in not requiring a targeted attack, meaning the AEA user cannot specify to attack the image to the selected category. In contrast, CW (Carlini & Wagner, 2017), FGM (Miyato, Dai, & Goodfellow, 2017), and JSMA (Papernot et al., 2016) have to formulate the category, and we randomly specify the category as 615. It is clear that all these methods effectively mislead the DL classification model without misleading the human visual perception, thereby satisfying Phenomenon (2). Within the context of DFKD, if the categories of the fake images identified by the instructor are used as real labels and the categories of the attacked fake images are used as soft labels, then not only is the lack of hard loss problem solved, but the process is also very similar to data-driven soft and hard loss, which is the motivation for us to introduce AEA.

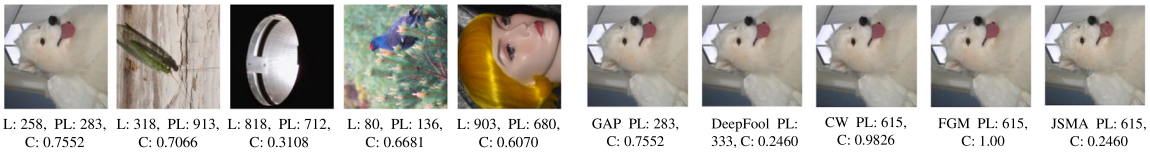
[3] The model used is ResNet18 and the dataset used is ImageNet. The true label and predicted label are denoted by L and PL, respectively, with C representing the confidence.

[4] We utilized pre-trained models like ResNet18 and ResNet34 to predict the CIFAR and ImageNet train datasets, achieving error rates between 0.01% and 0.04%.



(a) Correctly predicted scenario for real images (b) Incorrectly predicted scenario for real images

Fig. 4. The teacher's behavior of predicting different images.



(a) Attack effect of GAP (b) Attack effect of 5 AEA methods

Fig. 5. Predicted scenario for images that being attacked by AEA.

Table 2
Training rounds for GAP and attack effect.

Epoch	10	20	30	40	41	42	43	50
Attack state	Failed	Failed	Failed	Failed	Success	Success	Success	Success
Confidence	0.998	0.979	0.925	0.650	0.760	0.899	0.908	0.973

In this paper, GAP is selected to emulate the hard loss for two primary reasons. Firstly, GAP is a generative model-based approach that can be trained continuously with the DFKD process and takes less time. Secondly, GAP demonstrates proficiency in handling image classification tasks, as evidenced in Table 2. With the Adam optimizer, a batch size of 64 images, and a learning rate of 2e3, GAP achieves a successful attack in just 41 rounds. By the 50th epoch, the confidence level of the attacked image reaches an impressive 0.973. The effectiveness of the attack on GAP is further demonstrated through the visualization presented in Fig. 5(a), where it successfully deceives the classification model with a high confidence value.

GAP initializes a generative adversarial perturbation model gap and optimizes it using fake images and the teacher:

$$l_{gap} = -log(\text{CrossEntropy}(\widetilde{O}_t, \bar{y})), \tag{9}$$

where $\bar{y}$ is the least likely label of $\widetilde{x}$, as determined by the teacher. Executing the following equation is an attack :

$$\hat{x} = \widetilde{x} + gap(\widetilde{x}). \tag{10}$$

GAP also designs target versions. Please refer to Ref. Poursaeed et al. (2018) for further details and the structure of GAP. While AEA can easily replicate phenomena (1) and (2), additional design is required for (3) and (4). The confidence of x is overwhelmingly greater than that of the attacked x, but this is not the case for $\widetilde{x}$. During the pre-DFKD period, the $\widetilde{x}$ generated by RG is weak and far from the real image, which means that the confidence of $\widetilde{x}$ is likely to be lower than that of $\hat{x}$, failing to satisfy phenomenon (3). Therefore, we have designed an identity swap mechanism that let only the fake images with high confidence can be used for subsequent KD:

$$\hat{x}_i \leftrightarrow \widetilde{x}_i, \ \text{if} \ \hat{c}_i > \widetilde{c}_i \ \&\& \ \hat{y}_i == \widetilde{y}_i, \tag{11}$$

where $\leftrightarrow$ denotes the mutual exchange operation. The $\hat{c}_i$ and $\widetilde{c}_i$ represent the confidence level of $\hat{x}_i$ and $\widetilde{x}_i$ respectively, while $\hat{y}_i$ and $\widetilde{y}_i$ are the labels of $\hat{x}_i$ and $\widetilde{x}_i$ as determined by the teacher. The $\hat{y}_i == \widetilde{y}_i$ implies that the gap must have failed the attack to swap identities, in addition to the confidence. The reason for this design is to protect the diversity of the fake image. For phenomenon (4), the additional hyper-parameter pa is introduced:

$$\widetilde{x}_i = \begin{cases} \hat{x}_i, & \text{if} \ rand < pa \\ \widetilde{x}_i, & \text{otherwise} \end{cases}. \tag{12}$$

While it may be reasonable to set pa to 0.01%–0.04% based on phenomenon (4), this paper sets $pa = 0.1\%$ to account for the possibility of failed attacks. Note that in Eq. (12), only the attacked images are replaced, while the labels remain unchanged. At this point, the fake images have labels, all four phenomena are mimicked, and Eq. (4) becomes:

$$\begin{aligned} l_s = \alpha \cdot \tau^2 \cdot \text{KL}(\text{softmax}(\widetilde{O}_s/\tau), \text{softmax}(\widetilde{O}_t/\tau)) \\ +\beta \cdot \text{CrossEntropy}(\text{softmax}(\widetilde{O}_s, \widetilde{y})) \end{aligned}, \tag{13}$$

where α and β are hyper-parameters set to 0.9 and 0.1, respectively, following the conventional KD configuration.

4.4. Implementation of RG-HLS

The proposed RG-HLS aims to reuse part of the generator to reduce resource waste and simulate the hard loss of KD to improve classification accuracy. RG-HLS follows the general DFKD process, with detailed steps provided in Algorithm 1.

In Algorithm 1, θ_{gap} and θ_g represent the parameters of gap and g, respectively. Algorithm 1 applies two additional tricks: meta-learning and the warm-up mechanism. Meta-learning continuously optimizes the generator's input z, resulting in a batch of fake images that are stored in a collection to accelerate the algorithm's execution (Fang et al., 2022). The warm-up mechanism, controlled by the hyper-parameter e_{warmup}, prevents the student from training when the generator's performance is insufficient.

When training the model g, set the weight of l_{bn} in Eq. (8) to 10, the learning rate to 2e−3, the momentum to 0.9, and the weight decay factor to 1e−4, utilizing the Adam optimizer. For training the gap, the learning rate is also set to 2e−3, again employing the Adam optimizer. During the training of the student model, the parameters α and β in Eq. (13) are set to 0.9 and 0.1, respectively, aligning with vanilla KD. Notably, the hyperparameters for g are identical to those used in FM10, as FM10 has already determined suitable hyperparameters for the DFKD task addressed in this paper. For simplicity, the learning rate for training the gap is also set to be the same as that for training g. Additionally, other hyperparameters influencing the performance of the proposed method include kg and ks, which are set with reference to FM10. Among these hyperparameters, ep has the most significant impact on performance: an excessively large ep can result in an excessively long algorithm runtime, while an overly small ep may lead to inadequate training of g and the gap, affecting the strategy's effectiveness and potentially causing the student model to fail to converge fully. In this paper, the value of ep is set based on experience, prioritizing the convergence of the trainable part. For the specific values of these three hyperparameters, please refer to Section 5.1.

Algorithm 1 RG-HLS

Input: Pre-trained teacher t, g_1 and d, student s, mode

1: Initialize generative adversarial perturbation model gap;
2: **if** mode == Anew **then**
3: Initialize generative network g_1, g_2, Decoder d;
4: **else if** mode == Reuse **then**
5: Initialize generative network g_1;
6: Get g_2 and d published by the Anew mode task;
7: **end if**
8: $g \leftarrow (g_1, g_2, d)$; // Construction generator g
9: $\widetilde{xs} = \emptyset$; // Initializes the fake image set
10: **for** each $e \in [1, ep]$ **do**
11: **for** each $k \in [1, kg]$ **do**
12: $\widetilde{z}, \widetilde{\theta}_g \leftarrow Eq.\ (8)$; // Optimize g and z
13: $\theta_{gap} \leftarrow Eq.\ (9)$; // Optimize gap
14: **end for**
15: $z, \theta_g \leftarrow \widetilde{z}, \widetilde{\theta}_g$;
16: $\widetilde{x} \leftarrow Eq.\ (2)$; // Generate fake images
17: $\widetilde{xs} = \widetilde{xs} \cup \widetilde{x}$; // Fill the fake image set
18: **if** $e > e_{warmup}$ **then**
19: **for** each $k \in [1, ks]$ **do**
20: $\widetilde{x} \leftarrow sample(\widetilde{xs})$; // Randomly sample a batch of images from the fake image set
21: $\widetilde{y} \leftarrow t(\widetilde{x})$; // Query labels of fake images using teacher
22: $\hat{x} \leftarrow Eq.\ (10)$; // Adversarial example attack $\widetilde{x}$
23: $\widetilde{x}, \widetilde{y}, \hat{x}, \hat{y} \leftarrow Eq.\ (11)$;
24: $\widetilde{x} \leftarrow Eq.\ (12)$;
25: $s \leftarrow Eq.\ (13)$; // Optimize student s
26: **end for**
27: **end if**
28: **end for**
29: **if** mode == Anew **then**
30: $rg \leftarrow Eq.\ (7)$;
31: **return** s, rg, d
32: **else if** mode == Reuse **then**
33: **return** s
34: **end if**

During the initialization phase, the generator g and the generative adversarial perturbation model gap are initialized (line 1). If operating in Anew mode, all components of the generator are initialized. In Reuse mode, g_1 of the generator is initialized, and g_1 and d from previous Anew tasks are reused (lines 2–7). Upon entering the loop training framework, the general process is followed: in each epoch, the generator is iteratively trained kg times, followed by ks iterative training sessions for the student. Training g employs the meta-learning trick, and the gap is also trained at this time (lines 11–14). When the current epoch e exceeds ew (line 18), the AEA-based hard loss simulation is activated. First, a batch of images is sampled from the fake image set, and labels for these images are obtained from the teacher (lines 21–22). These images and labels are then attacked and updated (lines 23–24), followed by training the student (line 25). If the algorithm is in Anew mode, Eq. (7) is used to determine which generative network can be released for subsequent tasks (line 30). The trained s, rg, and d are then output (line 31). Otherwise, only s is output.

5. Experiments

This section presents the effectiveness of the proposed method from multiple perspectives, including comparisons with advanced DFKDs, safety considerations, and ablation experiments. Please note that the proposed RG-HLS is served for the image classification task, with all datasets utilized being specifically for image classification, and the comparison algorithms being exclusively or predominantly applicable to the image classification task.

5.1. Experimental setup

The α and β in Eq. (13) follow the vanilla KD experience (Hinton, Vinyals, & Dean, 2015) and are set to 0.9 and 0.1. In addition, this paper draws on the hyper-parameter configuration of FM (Fang et al., 2022), kg, and ks are set to 10 and 400, ep to 420, and the first 20 epochs are warm-up stages, that is, e_{warmup} is 20. For the Tiny-ImageNet dataset with higher image resolution, kg is set to 50. See our published code for other common hyper-parameters.

The experiments involve three widely recognized DL model architectures ResNet (He, Zhang, Ren, & Sun, 2016), VGG (Simonyan & Zisserman, 2014), WRN (Zagoruyko & Komodakis, 2016) in the KD domain.

- The VGG architecture, known for its very deep convolutional networks, extensively uses small 3 × 3 convolutional filters to increase the network's depth, achieving advanced results in image classification tasks.
- The ResNet architecture addresses the problem of network degradation through residual learning, which involves adding shortcut connections to the convolutional layers to prevent performance degradation as the network depth increases.
- The WRN architecture is a variant of ResNet that increases the number of convolutional kernels, thereby accelerating the computation during both training and inference.

We evaluated the accuracy of the proposed method on three well-known image classification datasets: CIFAR-10 (Krizhevsky, Hinton, et al., 2009), CIFAR-100 (Krizhevsky et al., 2009), MNIST (Lecun, Bottou, Bengio, & Haffner, 1998), SVHN (Netzer et al., 2011), and Tiny-ImageNet (Le & Yang, 2015). Both CIFAR-10 and CIFAR-100 contain 60,000 images of size 32 × 32, with 50,000 images in the training set and 10,000 in the validation set. They include 10 and 100 categories, respectively, with 5,000 and 500 images per category. The MNIST dataset comprises handwritten numbers contributed by 250 individuals, totaling 70,000 images. Specifically, it includes 60,000 images for the training set and 10,000 for the test set. The SVHN dataset is a real-world dataset of street signs, utilizing 73,257 images for training and 26,032 images for testing. Tiny-ImageNet comprises 200 categories, with 500 images per category in the training set and 50 in the validation set, and has a higher image resolution of 64 × 64. All RG-HLS experiments were conducted on computing devices equipped with NVIDIA A800 GPU, Intel(R) Xeon(R) Gold 6330 CPU @ 2.00 GHz, 256 GB Memory.

5.2. Comparison with advanced DFKDs

This section compares our proposed method with advanced DFKDs, including ADI (Yin et al., 2020), PRE-DFKD (Binici, Aggarwal, Pham, Leman, & Mitra, 2022), DAFL (Chen et al., 2019), ZSKT (Micaelli & Storkey, 2019), DFQ (Choi, Choi, El-Khamy, & Lee, 2020), RD-SKD (Han, Park, Wang, & Liu, 2021), STTSS (Wang, 2021), MAD (Do et al., 2022), MLISU (Patel et al., 2023), and FM10 (Fang et al., 2022). All comparison methods constitute DFKDs primarily or exclusively applicable to image classification tasks. In terms of fake image generation, ADI is based on random noise, whereas all other comparison methods utilize generators. Regarding their objectives, FM10 is dedicated to minimizing the training time required for DFKD. The remaining methods are focused on enhancing the classification accuracy of student models in data-free environments, aligning with the paper's purpose. Notably, FM10 has also demonstrated competitive classification accuracy results, so it is taken into account in this paper.

Table 3
Comparison on CIFAR datasets.

Dataset	Teacher Student	ResNet34 ResNet18	VGG11 ResNet18	WRN40_2 WRN16_1	WRN40_2 WRN40_1	WRN40_2 WRN16_2
CIFAR-10	Teacher's accuracy	95.70%	92.25%	94.87%	94.87%	94.87%
	Student's accuracy	95.20%	95.20%	91.12%	93.94%	93.95%
	ADI	93.26%	90.36%	83.04%	86.85%	89.72%
	DAFL	92.22%	81.10%	65.71%	81.33%	81.55%
	ZSKT	93.32%	89.46%	83.74%	86.07%	89.66%
	DFQ	94.61%	90.84%	86.14%	91.69%	92.01%
	FM10	94.05%	90.53%	89.29%	92.51%	**92.45%**
	PRE-DFKD	91.65%	87.26%	–	86.68%	83.57%
	STTSS	93.31%	–	–	–	–
	MLISU	94.02%	–	–	–	–
	MAD	**94.90%**	–	–	–	–
	RG-HLS (ours)	94.79%	**91.24%**	**89.36%**	93.18%	92.37
CIFAR-100	Teacher's accuracy	78.05%	71.32%	75.83%	75.83%	75.83%
	Student's accuracy	77.10%	77.10%	65.31%	72.19%	76.56%
	ADI	61.32%	54.13%	53.77%	61.33%	61.34%
	DAFL	74.47%	54.16%	20.88%	42.83%	43.70%
	ZSKT	67.74%	54.31%	36.33%	53.60%	54.59%
	DFQ	77.01%	66.21%	51.27%	54.43%	64.79%
	FM10	74.34%	67.44%	54.02%	63.91%	65.12
	PRE-DFKD	75.63%	**70.29%**	–	55.70%	49.54%
	MLISU	77.21%	–	–	–	–
	MAD	**77.31%**	–	–	–	–
	RG-HLS (ours)	74.13%	68.93%	**56.60%**	**67.40%**	**67.73%**

Table 4
Comparison on the Tiny-ImageNet dataset.

Teacher Student	ResNet34 ResNet18	VGG11 ResNet18	WRN40_2 WRN16_2
Teacher's accuracy	62.78%	56.59%	59.25%
Student's accuracy	64.53%	64.53%	56.44%
DFAD	57.34%	38.3%	33.07%
ADI	60.21%	46.67%	45.22%
DFQ	**62.44%**	48.33%	47.63%
MLISU	49.88%	–	–
RG-HLS (ours)	58.34%	**53.5%**	**47.86%**

The results are summarized in Tables 3–6. Note that for all KD tasks conducted by RG-HLS, g_2 and d are trained using the ResNet34-ResNet18 task. Teacher's accuracy and Student's accuracy denote the accuracy of the common method of training teacher and student. In some KD tasks, students outperformed teachers, and this was set up for task diversity reasons, as with other DFKDs (Choi et al., 2020; Fang et al., 2022; Yin et al., 2020). Those tables highlight the best results in bold, with the second-best results underlined. Tables 3 and 4 present the Top-1 accuracies, while Table 5 includes both Top-1 and Top-5 accuracies.

On CIFAR-10, RG-HLS generally outperforms other methods, especially in the WRN40_2-WRN40_1 task, where it significantly exceeds the comparison methods. RG-HLS outperformed ADI, DAFL, ZSKT, and PRE-DFKD, lagged behind MAD and DFQ on ResNet34-ResNet18, and was slightly behind FM10 on WRN40_2-WRN16_2. For CIFAR-100, RG-HLS performed significantly worse than PRE-DFKD and MAD on the ResNet34-ResNet18 and VGG11-ResNet18 tasks but showed very competitive results on the remaining three tasks. From a statistical perspective, among the 10 tasks listed in Table 3, RG-HLS achieved optimal performance on 5 tasks and sub-optimal performance on 4 tasks, surpassing all other compared methods. For the higher resolution Tiny-ImageNet, RG-HLS only lagged behind DFQ in the ResNet34-ResNet18 task but still ranked second and led the rankings for the other two tasks.

To achieve a more detailed analysis of the metrics, we provide the Top-1 and Top-5 accuracy results for the MNIST and SVHN datasets in Table 5.[5] For the MNIST dataset, both the proposed method and DAFL excel on the ResNet34-ResNet18 task, trailing slightly behind FM10 on the WRN40_2-WRN16_1 task. Across the remaining tasks, the proposed yield optimal performance. On the SVHN dataset, while RG-HLS falls 1.65 percentage points behind DAFL in Top-1 accuracy for the ResNet34-ResNet18 task, the disparity in Top-5 accuracy is negligible, at just 0.15%, marking it as sub-optimal. Similarly, for the WRN40_2-WRN40_1 task, RG-HLS lags behind FM10 by 1.55% in Top-1 accuracy and 0.3% in Top-5 accuracy, also falling into the sub-optimal category. For the WRN40_2-WRN16_1 task, our proposed method ranks third, with a minimal gap of 0.72% in Top-1 accuracy and just 0.17% in Top-5 accuracy compared to the top performer. For the remaining two tasks, the proposed method attains optimal results. From a more holistic perspective, the proposed method is comparable to FM10 on the SVHN dataset and surpasses FM10 on the remaining four datasets. Additionally, to gauge the categorical classification accuracy, we present the F1-scores for five methods on the VGG11-ResNet18-SVHN task in Table 6, where C1-C10 represent the 10 categories corresponding to images depicting numbers 1 through 10. Notably, RG-HLS achieves either optimal or sub-optimal results across all categories, with only C1 and C3 yielding sub-optimal outcomes, thereby showcasing its exceptional performance.

In conclusion, RG-HLS competently competes across all five real-world image classification datasets, affirming its robust generalization.

5.3. Security of the reusable generator

Other DFKDs do not consider reusing generators due to the risk of privacy leakage. Attackers may use the trained generator to generate fake images to attack DL models. To verify the security of the proposed reusable generator, we use the fake images generated by different data sources to perform the bit flip attack (BFA) (Rakin, He, & Fan, 2019). BFA is a data-driven attack method on DL models that records the gradient changes during model inference and flips the bits that

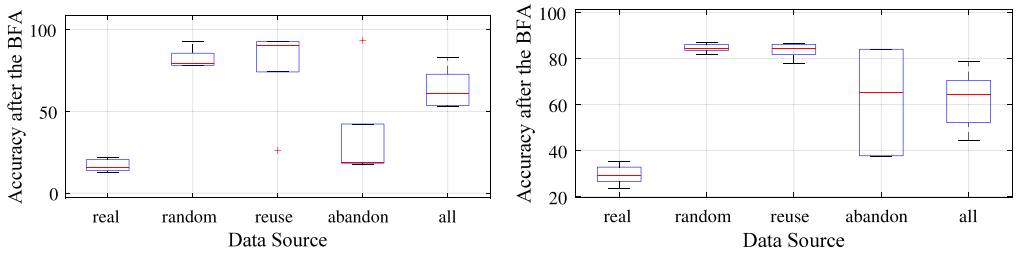
[5] All methods were executed locally, utilizing the publicly available code for FM10, DAFL, and RDSKD, and the code for ADI was replicated by Fang et al. (2022). The Top-5 accuracy metric for the MNIST dataset is not emphasized with bolding and underlining, as this accuracy consistently reaches 100% in most scenarios.

Table 5
Comparison on MNIST and SVHN datasets.

Dataset	Teacher	ResNet34		VGG11		WRN40_2		WRN40_2		WRN40_2	
	Student	ResNet18		ResNet18		WRN16_1		WRN40_1		WRN16_2	
MNIST	Teacher's accuracy	99.18%		99.63%		99.59%		99.59%		99.59%	
	Student's accuracy	99.17%		99.17%		99.62%		99.51%		99.64%	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
	FM10	99.13%	99.97%	99.48%	100.0%	**99.45%**	100.0%	99.56%	100.0%	99.49%	100.0%
	DAFL	**99.30%**	100.0%	99.17%	100.0%	99.23%	100.0%	99.30%	100.0%	99.38%	100.0%
	RDSKD	94.39%	100.0%	87.54%	99.02	76.13%	94.60%	88.10%	98.45%	91.20%	99.72%
	ADI	97.45%	99.95%	99.25%	100.0%	87.55%	99.14%	93.51%	99.88%	89.60%	99.88%
	RG-HLS(ours)	**99.30%**	100.0%	**99.62%**	100.0%	99.39%	100.0%	**99.60%**	100.0%	**99.50%**	100.0%
SVHN	Teacher's accuracy	95.57%		95.53%		95.90%		95.90%		95.90%	
	Student's accuracy	95.55%		95.55%		95.06%		95.86%		95.91%	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
	FM10	87.32%	98.34%	90.83%	**99.13%**	**90.09%**	98.76%	**92.66%**	**99.17%**	92.33%	99.15%
	DAFL	**94.81%**	**99.45%**	87.55%	98.74%	88.79%	98.73%	89.49%	98.87%	93.36%	99.37%
	RDSKD	89.59%	98.82%	89.60%	98.82%	89.62%	**98.92%**	89.91%	98.81%	90.62%	99.06%
	ADI	75.92%	96.19%	76.40%	96.81%	18.60%	60.88%	29.37%	75.25%	39.50%	80.74%
	RG-HLS(ours)	93.16%	99.30%	**91.32%**	98.97%	89.37%	98.59%	91.11%	98.87%	**92.37%**	**99.16%**

Table 6
Category-by-category F1-scores of 5 methods on the VGG11-ResNet18-SVHN task.

Methods	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
FM10	**89.90%**	92.50%	**94.70%**	86.90%	92.40%	90.20%	88.00%	91.90%	86.70%	88.40%
DAFL	86.53%	90.41%	91.28%	85.19%	87.41%	88.43%	85.45%	89.34%	77.77%	85.26%
RDSKD	87.41%	90.62%	93.99%	86.22%	89.80%	91.60%	88.41%	90.48%	84.73%	86.07%
ADI	50.87%	81.26%	87.50%	67.78%	76.90%	79.40%	65.88%	83.12%	70.95%	70.54%
RG-HLS(ours)	87.59%	**93.23%**	94.42%	**87.49%**	**93.27%**	**91.92%**	**89.91%**	**92.66%**	**87.51%**	**89.25%**



(a) ReNet34 (b) VGG11

Fig. 6. Teachers' accuracy subjected to the BFA on CIFAR-10.

cause the greatest loss. The BFA configuration used was the original one (Rakin et al., 2019), and the accuracy of the attacked model was recorded after flipping 20 bits, as shown in Fig. 6.

The terms "real" and "random" indicate whether the data driving the BFA comes from the real dataset or random data, respectively. "Reuse" and "abandon" refer to the data sources generated by reusable and non-reusable generative networks, respectively, as outputted by Eq. (7). "all" refers to the data sources generated by reusing all three components, g_1, g_2, and d. "Accuracy" refers to the image classification accuracy of the model after an attack. Please note that this paper reports the attack results on the ResNet34 and VGG11 models, whose accuracy on CIFAR-10 before being attacked was 95.70% and 92.25%, respectively.

Fig. 6 shows that the real dataset results in the BFA that reduces the models' accuracy to close to random guessing, while random images are almost not effective in driving the BFA. The generative network dropped by Eq. (7) produces the best BFA results outside of "real", followed by "all". We have determined that Reuse–the published generative network, which is almost the same as "random", cannot successfully execute a BFA attack. In summary, the reusable generator designed in this paper ensures the security of DFKD.

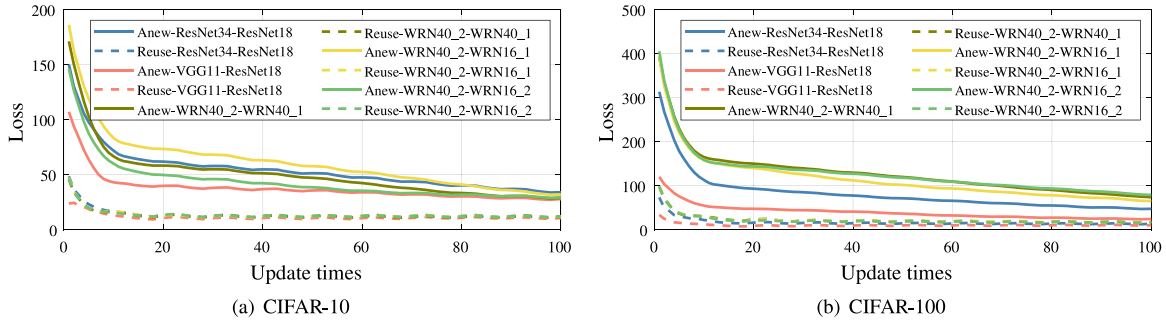
Table 7
Comparison of Anew mode and 3 Reuse modes on CIFAR-100 dataset.

Teacher	VGG11	WRN40_2	WRN40_2	WRN40_2
Student	ResNet18	WRN16_1	WRN40_1	WRN16_2
Anew	68.91	54.32	65.24	65.93
Reuse-finetune	68.20	52.27	65.01	65.26
Reuse-fix	67.92	52.28	64.72	65.11
Reuse	**69.26**	**56.13**	**65.71**	**67.20**

5.4. Effect of the reusable generator

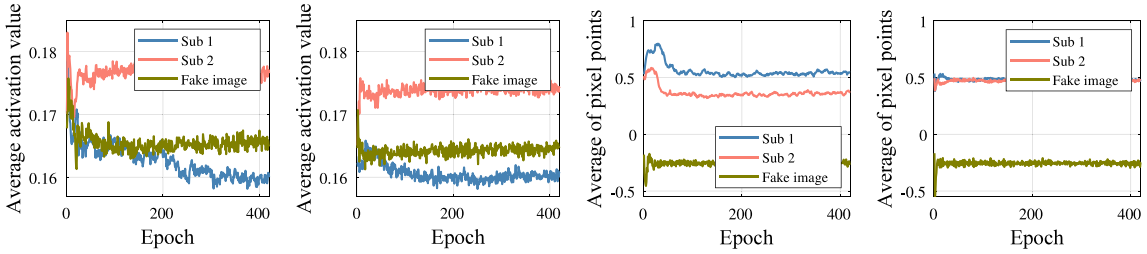
To verify the effectiveness of the proposed reusable generator, we present the results of multiple runs of the rg configurations in Table 7. In the Reuse-finetune configuration, g_1 is trained at the original learning rate, lr_g, while g_2 and d are trained at a learning rate of $lr_g * 0.1$. In contrast, Reuse-fix trains only g_1. The results in Table 7 show that the best accuracy is obtained in the Reuse configuration (i.e., the Reuse mode and the Anew model are trained in the same way).

Intuitively, Reuse-fix or Reuse-finetune should perform better than Reuse because they align more closely with the pre-training and fine-tuning process. However, the results in Table 7 contradict this intuition. This discrepancy arises for two reasons. Firstly, although Reuse-fix



(a) CIFAR-10 (b) CIFAR-100

Fig. 7. Convergence for Anew and Reuse modes.



(a) Average of pixel points (Reusable generator) (b) Average of pixel points (Without decoder) (c) Average activation value (Reusable generator) (d) Average activation value (Without decoder)

Fig. 8. Metrics curve with and without decoder.

reuses a generative network and decoder, the reused parts are no longer trained, placing the entire fitting burden on the generative network that has not been reused, which can lead to overfitting. Secondly, for deep learning, the learning rate for general de novo training is typically around 1e−1, while for fine-tuning, it is around 1e−2. For stability optimization purposes, however, the hyper-parameter lr_g used to optimize the generator is low enough (less than 1e−3). In the Reuse-finetune mode, the learning rate for the reused parts is reduced by an order of magnitude, preventing them from being fully optimized.

Fig. 7 depicts the loss variation of rg on CIFAR-10 and CIFAR-100 for the Anew and Reuse modes, focusing on the initial 100 updates for clarity. Compared to the Anew mode, the initial loss of the Reuse mode is significantly lower and can rapidly converge to a lower loss level, regardless of the dataset and the DFKD task. The above demonstrates that rg not only improves accuracy but also expedites convergence for the generator.

Fig. 8 presents a comparative analysis of the sub-feature maps, namely Sub 1 and Sub 2, produced by two sub-generative networks in reusable generator, and the final fake images. This figure highlights the disparities between the reusable generator and its without decoder variant. The average of pixel points refers to the mean of values within the sub-images, while average activation value denotes the mean of feature maps generated prior to the fully connected layer, during the teacher's reasoning process over the image. A noteworthy observation is that the pixel averages of the two sub-feature maps are remarkably similar in the absence of a decoder (Fig. 8(b)), indicating a potential overlap or high degree of similarity between the two sub-generative networks. This underscores the significance of the decoder, which not only facilitates the fusion of the two sub-feature maps but also ensures that a given sub-generative network does not same with another sub-generative network. In addition, the lack of decoder led to the decrease of the average activation difference between sub-feature maps and fake's feature maps (Fig. 8(d)), and at this time, a sub-generator network is more likely to degenerate into a noise generator. The experiment in Fig. 8 employs 256 images and their sub-features per epoch, utilizing ResNet34 as the teacher.

Table 8
Comparison of AEA-based hard loss strategy and hard loss only case on CIFAR-100 dataset.

Teacher Student	VGG11 ResNet18	WRN40_2 WRN16_1	WRN40_2 WRN40_1	WRN40_2 WRN16_2
Reuse	**69.26**	56.13	65.71	67.20
Reuse+HL	68.81	56.37	65.55	66.99
Reuse+HL+AEA	68.93	**56.60**	**67.40**	**67.73**

5.5. Effect of the AEA-based hard loss simulation

The AEA-based hard loss simulation aims to provide the DFKD paradigm with a hard loss component, which is crucial in data-driven KD. To verify the effectiveness of this designed strategy, ablation experiment is conducted in this section. Table 8 compares the accuracy rates of three scenarios: using only the Reuse mode, introducing teacher-calibrated labels and constructing hard loss (Reuse + HL) on top of the Reuse mode, and implementing AEA simulated data-driven KD on top of Reuse + HL.

The results indicate that the addition of hard loss alone does not yield significant improvements and even leads to a decline in accuracy compared to the Reuse mode. This is due to the fact that the hard loss is entirely determined by the teacher. Ideally, the teacher would not correctly predict all the samples, meaning the real hard loss cannot be fully controlled by the teacher. However, the introduction of AEA significantly boosts the accuracy, indicating that the AEA-HL design employed in this paper effectively simulates this ideal situation.

5.6. Discussion of the generator with multiple generative networks

The generator of RG-HLS includes two generative networks. This configuration could lead to the misconception that performance improvements are solely due to an increased number of generative networks. To clarify this, we explore generators with varying numbers of generative networks in this section. Fig. 9 illustrates the classification accuracy achieved by generators with 1 to 7 generative networks in both Anew and Reuse modes. To enhance the generalizability of the

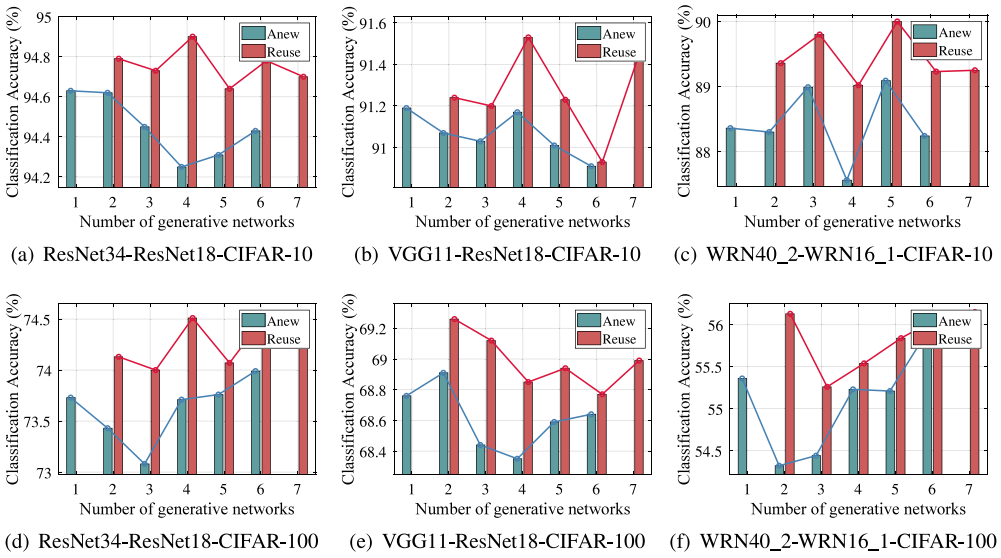


Fig. 9. Accuracy of generators with different number of generative networks.

Table 9
Overhead of the trainable part with different number of generative networks.

Number of generative networks	1	2	3	4	5	6	7
FLOPs	1.74E+11	2.04E+11	2.34E+11	2.6382e+11	2.94E+11	3.25E+11	3.55E+11
Paras	1.35E+07	1.58E+07	1.82E+07	2.05E+07	2.28E+07	2.52E+07	2.75E+07
Train time (s)	9.35	13.25	16.34	20.95	19.00	19.50	24.12
Inference time (s)	8.67E−03	1.11E−02	1.60E−02	2.16E−02	2.47E−02	7.47E−02	3.58E−02

conclusions, the experiment encompassed three teacher–student pairs and utilized two datasets: CIFAR-10 and CIFAR-100. In the Reuse mode, the generator always reuses previously trained generative networks from the Anew mode. For instance, a generator with 3 generative networks (denoted as generator-3-Reuse) reuses two generative networks that have been trained after completing the generator-2-Anew task, while the remaining one is randomly initialized.

Upon observing Fig. 9, it becomes evident that there is no direct correlation between the number of generative networks and classification accuracy for either mode. For instance, in the case of the VGG11-ResNet-CIFAR-10 task in Reuse mode, the optimal classification accuracy is achieved with 4 generative networks, whereas in Anew mode, it is achieved with just one. For the WRN40_2-WRN16_1-CIFAR-100 task, the optimal configurations are 2 and 6 generative networks, respectively. This is in line with the intuition that more generative models imply more trainable parameters, and that too many trainable parameters do not directly lead to performance gains. Another notable observation is that when both models possess the same number of generative networks, the Reuse model consistently outperforms the Anew model, which is consistent with our design expectations. These two phenomena demonstrate that the performance improvement of RG-HLS does not stem from increasing the number of generative networks, but rather from reusing the trained generative networks. Furthermore, as the number of generative networks escalates, the cost of training reusable generators increases significantly.

Table 9 lists the running overhead of the trainable part with different numbers of generative networks. Here, FLOPs measure the computational complexity of the generator and the student, i.e., the number of floating-point operations needed to process one input. Paras measure the storage requirements of the model, i.e., the total number of parameters that need to be trained and stored. Train time and Inference time represent the time required for a generator to train for one epoch and generate a batch (256) of fake images, respectively. As the number of generative networks increases, both FLOPs and Paras increase essentially linearly. Regarding inference time, the generator takes very little time regardless of the number of networks. However, for training time, the duration increases significantly when the number of generative networks reaches three.

The aforementioned analysis can be condensed into three key conclusions:

- The classification accuracy does not exhibit a direct correlation with the number of generative networks.
- The efficacy of the reusable generator stems from its ability to reuse the generative network, rather than from increasing the number of generative networks.
- An increase in the number of generative networks results in a substantial rise in the training cost of a reusable generator network.

Furthermore, regardless of the number of generative networks, the variation in their classification accuracy remains minimal, typically not exceeding 1%. Consequently, we configure the reusable generator with two generative networks, enabling it to provide the advantages of the reuse model without significantly augmenting the training overhead.

6. Conclusion

To address the challenges posed by non-reusable generators and the limitations of hard loss in DFKDs, this paper introduces the reusable generator and AEA-based hard loss strategy. The reusable generator is designed to maximize the utilization of generative models trained through the KD process while maintaining rigorous data privacy and security. The AEA is introduced to improve classification accuracy. Extensive experimental results validate that our proposed strategy aligns with our design expectations.

There is no free lunch, however, the proposed method is not devoid of limitations. The most prominent limitation lies in the absence of generalization, as RG-HLS employs the cross-entropy loss, suitable primarily for classification tasks, without developing customized loss functions for diverse applications. Furthermore, the incorporation of

the AEA method, which is similarly well-suited for classification, exacerbates this issue. To address this, our next endeavor involves exploring various generative models, including diffusion models, and devising appropriate training methodologies to enhance generalizability and extend applicability to a broader range of tasks, such as objective detection and semantic segmentation.

CRediT authorship contribution statement

Yafeng Sun: Methodology, Software and Writing – original draft. **Xingwang Wang:** Conceptualization, Methodology, Writing – review & polishing. **Junhong Huang:** Supervision, Project administration. **Shilin Chen:** Proofreading. **Minghui Hou:** Data curation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

I have shared the link to my code in the manuscript.

References

Binici, K., Aggarwal, S., Pham, N. T., Leman, K., & Mitra, T. (2022). Robust and resource-efficient data-free knowledge distillation by generative pseudo replay.

Carlini, N., & Wagner, D. (2017). Towards evaluating the robustness of neural networks. In *2017 IEEE symposium on security and privacy* (pp. 39–57). IEEE.

Chen, H., Wang, Y., Xu, C., Yang, Z., Liu, C., Shi, B., et al. (2019). Data-free learning of student networks. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 3514–3522).

Chen, W., Xuan, Y., Yang, S., Xie, D., Lin, L., & Zhuang, Y. (2024). Better together: Data-free multi-student coevolved distillation. *Knowledge-Based Systems*, *283*, Article 111146.

Choi, Y., Choi, J., El-Khamy, M., & Lee, J. (2020). Data-free network quantization with adversarial knowledge distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops* (pp. 710–711).

Do, K., Le, T. H., Nguyen, D., Nguyen, D., Harikumar, H., Tran, T., et al. (2022). Momentum adversarial distillation: Handling large distribution shifts in data-free knowledge distillation. *Advances in Neural Information Processing Systems*, *35*, 10055–10067.

Fang, G., Mo, K., Wang, X., Song, J., Bei, S., Zhang, H., et al. (2022). Up to 100x faster data-free knowledge distillation. *vol. 36*, In *Proceedings of the AAAI conference on artificial intelligence* (pp. 6597–6604).

Fang, G., Song, J., Wang, X., Shen, C., Wang, X., & Song, M. (2021). Contrastive model inversion for data-free knowledge distillation. arXiv preprint arXiv:2105.08584.

Gou, J., Chen, Y., Yu, B., Liu, J., Du, L., Wan, S., et al. (2024). Reciprocal teacher-student learning via forward and feedback knowledge distillation. *IEEE Transactions on Multimedia*, *26*, 7901–7916. http://dx.doi.org/10.1109/TMM.2024.3372833.

Gou, J., Hu, Y., Sun, L., Wang, Z., & Ma, H. (2024). Collaborative knowledge distillation via filter knowledge transfer. *Expert Systems with Applications*, *238*, Article 121884.

Gou, J., Sun, L., Yu, B., Wan, S., & Tao, D. (2023). Hierarchical multi-attention transfer for knowledge distillation. *ACM Transactions on Multimedia Computing, Communications, and Applications*, *20*(2), http://dx.doi.org/10.1145/3568679.

Guo, W., Li, X., & Shen, Z. (2024). A lightweight residual network based on improved knowledge transfer and quantized distillation for cross-domain fault diagnosis of rolling bearings. *Expert Systems with Applications*, *245*, Article 123083.

Guo, J., Zhan, W., Jiang, Y., Ge, W., Chen, Y., Xu, X., et al. (2024). MFHOD: Multi-modal image fusion method based on the higher-order degradation model. *Expert Systems with Applications*, *249*, Article 123731.

Han, S., Lin, C., Shen, C., Wang, Q., & Guan, X. (2023). Interpreting ddversarial examples in deep learning: A review. *ACM Computing Surveys*.

Han, P., Park, J., Wang, S., & Liu, Y. (2021). Robustness and diversity seeking data-free knowledge distillation. In *ICASSP 2021-2021 IEEE international conference on acoustics, speech and signal processing* (pp. 2740–2744). IEEE.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778).

Heo, B., Kim, J., Yun, S., Park, H., Kwak, N., & Choi, J. Y. (2019). A comprehensive overhaul of feature distillation. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 1921–1930).

Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531.

Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images.

Le, Y., & Yang, X. (2015). Tiny imagenet visual recognition challenge. *CS 231N*, *7*(7), 3.

Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324. http://dx.doi.org/10.1109/5.726791.

Li, Y., Li, P., Yan, D., Liu, Y., & Liu, Z. (2024). Deep knowledge distillation: A self-mutual learning framework for traffic prediction. *Expert Systems with Applications*, Article 124138.

Li, Z., Li, X., Yang, L., Zhao, B., Song, R., Luo, L., et al. (2023). Curriculum temperature for knowledge distillation. *vol. 37*, In *Proceedings of the AAAI conference on artificial intelligence* (pp. 1504–1512).

Li, H., & Wu, X.-J. (2018). DenseFuse: A fusion approach to infrared and visible images. *IEEE Transactions on Image Processing*, *28*(5), 2614–2623.

Li, J., Zhou, S., Li, L., Wang, H., Bu, J., & Yu, Z. (2023). Dynamic data-free knowledge distillation by easy-to-hard learning strategy. *Information Sciences*, *642*, Article 119202.

Luo, L., Sandler, M., Lin, Z., Zhmoginov, A., & Howard, A. (2020). Large-scale generative data-free distillation. arXiv preprint arXiv:2012.05578.

Ma, D., Zhang, K., Cao, Q., Li, J., & Gao, X. (2024). Coordinate attention guided dual-teacher adaptive knowledge distillation for image classification. *Expert Systems with Applications*, *250*, Article 123892.

Macas, M., Wu, C., & Fuertes, W. (2024). Adversarial examples: A survey of attacks and defenses in deep learning-enabled cybersecurity systems. *Expert Systems with Applications*, *238*, Article 122223. http://dx.doi.org/10.1016/j.eswa.2023.122223.

Micaelli, P., & Storkey, A. J. (2019). Zero-shot knowledge transfer via adversarial belief matching. *Advances in Neural Information Processing Systems*, *32*.

Miyato, T., Dai, A. M., & Goodfellow, I. (2017). Adversarial training methods for semi-supervised text classification. In *International conference on learning representations*. URL: https://openreview.net/forum?id=r1X3g2_xl.

Moosavi-Dezfooli, S.-M., Fawzi, A., & Frossard, P. (2016). DeepFool: A simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

Mukherjee, D. (2023). Detection of data-driven blind cyber-attacks on smart grid: A deep learning approach. *Sustainable Cities and Society*, *92*, Article 104475.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A. Y., et al. (2011). Reading digits in natural images with unsupervised feature learning. *vol. 2011*, In *NIPS workshop on deep learning and unsupervised feature learning* (p. 4). Granada, 2.

Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z. B., & Swami, A. (2016). The limitations of deep learning in adversarial settings. In *2016 IEEE European symposium on security and privacy* (pp. 372–387). IEEE.

Patel, G., Mopuri, K. R., & Qiu, Q. (2023). Learning to retain while acquiring: combating distribution-shift in adversarial data-free knowledge distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 7786–7794).

Poursaeed, O., Katsman, I., Gao, B., & Belongie, S. (2018). Generative adversarial perturbations. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4422–4431).

Rakin, A. S., He, Z., & Fan, D. (2019). Bit-flip attack: Crushing neural network with progressive bit search. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 1211–1220).

Shao, R., Zhang, W., & Wang, J. (2023). Conditional pseudo-supervised contrast for data-free knowledge distillation. *Pattern Recognition*, *143*, Article 109781.

Shao, R., Zhang, W., Yin, J., & Wang, J. (2023). Data-free knowledge distillation for fine-grained visual categorization. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 1515–1525).

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

Tang, L., Xiang, X., Zhang, H., Gong, M., & Ma, J. (2023). DIVFusion: Darkness-free infrared and visible image fusion. *Information Fusion*, *91*, 477–493.

Wang, Z. (2021). Data-free knowledge distillation with soft targeted transfer set synthesis. *vol. 35*, In *Proceedings of the AAAI conference on artificial intelligence* (pp. 10245–10253). 11.

Wang, Y., Qian, B., Liu, H., Rui, Y., & Wang, M. (2024). Unpacking the gap box against data-free knowledge distillation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Wang, X., Sun, Y., Chen, X., & Xu, H. (2024). DDEP: Evolutionary pruning using distilled dataset. *Information Sciences*, *659*, Article 120048. http://dx.doi.org/10.1016/j.ins.2023.120048, URL: https://www.sciencedirect.com/science/article/pii/S0020025523016341.

Xie, R., Chen, Z., Wu, C., & Li, T. (2024). PPFGED: Federated learning for graphic element detection with privacy preservation in multi-source substation drawings. *Expert Systems with Applications*, *243*, Article 122758.

Yang, M., Tang, J., Dang, S., Chen, G., & Chambers, J. A. (2024). Multi-distribution mixture generative adversarial networks for fitting diverse data sets. *Expert Systems with Applications*, Article 123450.

Yilmaz, B., & Korn, R. (2024). A comprehensive guide to generative adversarial networks (GANs) and application to individual electricity demand. *Expert Systems with Applications*, *250*, Article 123851.

Yin, H., Molchanov, P., Alvarez, J. M., Li, Z., Mallya, A., Hoiem, D., et al. (2020). Dreaming to distill: Data-free knowledge transfer via deepinversion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 8715–8724).

Yu, S., Chen, J., Han, H., & Jiang, S. (2023). Data-free knowledge distillation via feature exchange and activation region constraint. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 24266–24275).

Zagoruyko, S., & Komodakis, N. (2016). Wide residual networks. arXiv preprint arXiv: 1605.07146.

Zhang, Y., Wu, M., Tian, G. Y., Zhang, G., & Lu, J. (2021). Ethics and privacy of artificial intelligence: Understandings from bibliometrics. *Knowledge-Based Systems*, *222*, Article 106994.

Zhao, B., Cui, Q., Song, R., Qiu, Y., & Liang, J. (2022). Decoupled knowledge distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 11953–11962).

Zhao, H., Sun, X., Dong, J., Yu, H., & Wang, G. (2022). Multi-instance semantic similarity transferring for knowledge distillation. *Knowledge-Based Systems*, *256*, Article 109832.

Zhao, Z., Xu, Y., Li, Y., Zhao, Y., Wang, B., & Wen, G. (2023). Sparse actuator attack detection and identification: A data-driven approach. *IEEE Transactions on Cybernetics*.

Zhou, T., Li, Q., Lu, H., Cheng, Q., & Zhang, X. (2023). GAN review: Models and medical image fusion applications. *Information Fusion*, *91*, 134–148.