IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 31 March 2023, accepted 3 May 2023, date of publication 8 May 2023, date of current version 17 May 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3273952

RESEARCH ARTICLE

Deep Generative Knowledge Distillation by Likelihood Finetuning

JINGRU LI[1], XIAOFENG CHEN[2,3], PEIYU ZHENG[2], QIANG WANG[3], AND ZHI YU[1]
[1]EAGLE Laboratory, Zhejiang University, Hangzhou 310058, China
[2]Hangzhou Qulian Technology Company Ltd., Hangzhou 310058, China
[3]Blockchain Research Center, Zhejiang University, Hangzhou 310058, China

Corresponding author: Zhi Yu (yuzhirenzhe@zju.edu.cn)

This work was supported in part by the Key Research and Development Program of Zhejiang Province under Grant 2021C01105, in part by the Key Research and Development Program of Guangdong Province under Grant 2020B0101090003, and in part by the National Key Research and Development Program of China under Grant 2021YFB2701100.

ABSTRACT Knowledge Distillation (KD) is designed to train smaller student models using a larger pretrained teacher model. However, in decentralized data systems such as blockchain, privacy concerns may arise, making the data inaccessible. To address this issue, Data-Free KD (DFKD) methods have been proposed, which extract prior knowledge from teacher networks and use it to synthesize data for KD. Previous DFKD methods faced challenges due to the large search space of data generation. Recently, deep generative models (DGMs) have been proposed to learn data distribution using deep networks, which provides an efficient way to reduce the search space by generating a set of pseudo data. In this paper, we explore the performance of KD trained using pseudo samples generated by pretrained DGMs and find that the correlation with image quality is not always positive. Based on this observation, we propose a new DFKD framework called Generative Knowledge Distillation (GenKD) that reduces the search space by constructing a prior distribution modeled by DGMs for their power of likelihood estimation. Specifically, we use energy-based models (EBM) to generate data from the Maximum Likelihood Estimation (MLE) of the EBM and gradients from downstream KD tasks by policy gradient. We then train the student model using the pretrained teacher model and pseudo samples. We also implement our GenKD framework on several widely-used benchmarks, including CIFAR100, CIFAR10, and SVHN. Our experiments demonstrate that we can generate high-quality pseudo samples quantitatively and qualitatively using GenKD. Additionally, the top-1 accuracy of the student network can approach state-of-the-art (SOTA) DFKD methods trained using fewer pseudo samples and less generation time.

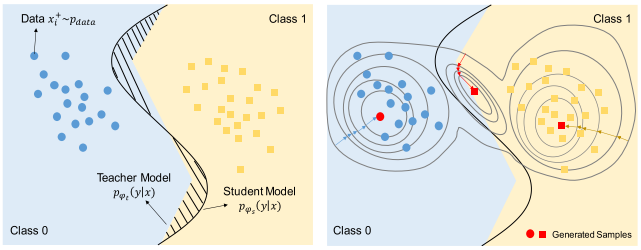
INDEX TERMS Knowledge distillation, deep generative model, image quality evaluation, data-free knowledge distillation.

I. INTRODUCTION

Recently, Deep Neural Networks (DNNs) have gained widespread adoption across numerous machine learning applications, and they often suffer from issues of complexity and portability, particularly when dealing with limited resources. In response to this, Knowledge Distillation (KD) [1], [2], [3], [4] offers a convenient solution for training smaller student networks using a pretrained teacher model, which minimizes the divergence between the teacher and student models. Within the field of blockchain, deep networks are often necessary for proof of work (PoW), and KD can serve to accelerate PoW for smaller networks.

In recent years, the blockchain system has gained increasing attention due to its ability to ensure privacy protection [5] and support for federal learning [6]. However, the practical application of blockchain technology often involves inaccessible training data, which limits the ability of student networks to learn explicit logit information from the teacher network for effective training. To address this challenge,

The associate editor coordinating the review of this manuscript and approving it for publication was Zhe Xiao.



(a) Previous DFKD methods. The black shadow represents the interest in useful data.

(b) Our GenKD framework. The contour represents the learned distribution by DGMs.

FIGURE 1. The main idea of GenKD. For the previous DFKD framework, they searched the whole data space. Our GenKD model can search data by the path of the Boltzmann distribution and also promises MLE. Better viewed on screen.

Data-Free Knowledge Distillation (DFKD) has been proposed as an approach to extracting knowledge from a pretrained teacher model. DFKD employs various loss functions to evaluate the quality of the generated data, referred to as **pseudo samples**. Previous work on DFKD [7], [8], [9], [10], [11] have employed different strategies to design loss functions and generate data that represent knowledge. For instance, some studies [8], [9], [12], [13] use a generator to produce data points that are similar to the real data points, while others [10], [11], [14] adopt adversarial learning [15], [16], [17] to generate challenging data points that are beneficial for training the student network. Recently, DeepInversion [10] has been proposed to align the distribution of each layer using the running mean and variance of batch normalization layer [18]. However, such an alignment approach is not explicit enough and may not efficiently search for useful data for KD training, as it generates pseudo samples instance by instance. We hope to find a distribution-level alignment for the generation of pseudo samples and KD training.

One question is whether generating pseudo samples that closely resemble the data distribution can improve KD performance. To address this, we conducted experiments using a variety of pretrained generative models to generate pseudo samples and explored the relationship between the likelihood estimation of the generated distribution and KD performance to determine the target of the generation stage. To efficiently search for informative data points for knowledge distillation, we developed a simple method for modeling the distribution of multiple deep generative models (DGMs) using prior distribution information. We evaluated the quality of the pseudo samples generated by DGMs optimized by Maximum Likelihood Estimation (MLE), such as flow-based models [19], [20], Energy-Based Models (EBMs) [21], [22], [23], [24], diffusion models [25], [26], [27], [28], and score-based models [29], [30], [31], and examined their relationship with KD performance. Despite the success of DGMs in likelihood and data estimation, estimating the discriminative model $p(y|\mathbf{x})$ remains a challenge for most KD tasks. The results of our research provide important insights into the potential of using pseudo samples generated by DGMs to improve knowledge distillation and offer a starting point for future work on the development of more effective and accurate knowledge distillation methods.

Motivated by previous work on DFKD, we propose a novel framework called Generative Knowledge Distillation (GenKD) that leverages the power of DGMs to improve the data distribution approximation and search process for knowledge distillation. Unlike previous DFKD approaches, which search the entire data space based on guidance from the teacher model, GenKD efficiently finds informative data points by utilizing the Boltzmann distribution as a guide. To achieve faster identification of hard samples, we use Langevin dynamics to gradually approach the Maximum Likelihood Estimation (MLE) of Energy-Based Models (EBMs). This approach enables GenKD to traverse the steepest path of the energy function (gray contours) and dynamically locate the MLE of class-conditional distribution while finding hard samples for KD training. As illustrated in Figure 1b, the proposed search process is both interpretable and computationally efficient, and can be extended to other MLE-based DGMs, such as DDPM, by incorporating likelihood function items during training. We believe that the GenKD framework has the potential to significantly improve the performance of knowledge distillation by leveraging the power of DGMs. This framework offers a promising approach for addressing the challenges of identifying informative data points and accurately approximating the data distribution in KD tasks and can be applied to a wide range of DGMs for different types of KD tasks. As the DGMs are black-box for the user to train KD, the generated pseudo samples by GenKD only contain information that is beneficial to the KD training. Therefore, GenKD does not have extra privacy leakage while achieving pseudo samples.

Our contributions of this work are:

- We first build a DGM module to learn the distribution of original data from pretrained teacher network. It provides a better estimation of data distribution and teacher models.
- We give some practical evidence on the relationship between the generation quality, estimation likelihood of DGMs and the performance of KD, and we find that the generation quality is not always positively correlated to the KD performance.
- As an example of GenKD, we update the likelihood function by the gradient from the downstream task by the policy gradient algorithm in EBM, which is effective for training KD.
- Experiments show that GenKD efficiently generates high-quality pseudo samples. Our GenKD can help the student model to learn KD on different benchmarks if we provide a better DGM, and we also explore the effect of image quality and class-conditional distribution on the DFKD task.

This paper is organized as follows. Section II describes the related work of DFKD and DGMs. Section III describes

the effect of DGM on the KD and the relationship between image quality and KD performance. In section IV, we take EBM as an example to construct a KD method that uses data samples with finetuned log-likelihood sampling. In section V, we discuss the mean idea of GenKD related to previous methods. In section VI, we provide some experiments of our GenKD on CIFAR10, CIFAR100, and SVHN, and in section VII we give a conclusion of this paper.

II. RELATED WORK

A. KNOWLEDGE DISTILLATION AND DATA-FREE KNOWLEDGE DISTILLATION

Knowledge distillation (KD) [1] was originally designed to enable smaller neural networks to learn from a larger, pre-trained teacher model. By using the ''soft labels'' or logits produced by the teacher model, the student model can learn not only from the ground truth but also from the knowledge embedded in the teacher model. In recent work, various approaches have been proposed to improve KD, such as aligning logits [32], [33], [34], [35], [36], [37] or intermediate feature layers [38], [39], [40], [41], [42], [43], [44] between the teacher and student models.

DFKD, also known as zero-shot knowledge distillation, refers to the technique of training a knowledge distillation framework without access to the original training data [7], [45], [46]. This is achieved by directly extracting knowledge from a pre-trained teacher model to recover the knowledge from the data. For example, Nayak et al. [7] extract knowledge from a pre-trained teacher model by minimizing the classification error of generated samples, while Chen et al. [13] learn with the assistance of prior knowledge from unlabeled extensive benchmarks like ImageNet [47].

Other methods, like [10], [11], and [14], focus on generating image samples with adversarial or contrastive methods. The primary goal is to visualize the knowledge from pre-trained deep networks using high-quality pseudo samples. For example, Yin et al. [10] propose DeepInversion and Adaptive DeepInversion (ADI), which capture the response from teacher networks and directly update in the image space beginning with random or uniform samples. These methods align the mean and variation of each batch normalization layer of the teacher model to extract knowledge from the pre-trained teacher distribution. Recently, some work have proposed using pseudo samples searched from previous epochs to avoid the catastrophic forgetting problem in DFKD [48], [49].

However, searching the entire data space $\mathcal{X}$ is computationally expensive, and simply setting a distribution through a generator may not result in an optimal data prior. Adopting a probabilistic perspective, as in ADI [10], can improve the realism and robustness of generated pseudo samples.

B. DEEP GENERATIVE MODELS

Deep generative models (DGMs) utilize probabilistic models, which are parameterized by deep neural networks, to describe the distribution of the data space. They have been extensively used for various applications such as image generation [25], [26], [50], online learning [51], text generation [52], text-to-image generation reinforcement learning [51] and out-of-distribution detection [16], [22], [50]. Maximum Likelihood Estimation (MLE) is typically used for optimizing DGMs.

Variational Autoencoders (VAEs) [53], [54], [55], [56] optimize the DGM by marginalizing the distribution of low-dimensional latent variables. Flow-based models [19], [20] optimize MLE by log determinants of latent variables combined with a chain of latent variables with the same dimension. Energy-Based Models (EBMs) [21], [22], [23], [24] explicitly model the distribution of data, which theoretically promises a more realistic result, and they use Markov Chain Monte Carlo (MCMC) [57], [58], [59], [60] for sampling. Deep diffusion models [25], [26], [27], [28] define a chain of Markov processes and optimize the likelihood by the approximation of variational lower bound in every single process. The score-based model [31], [61] defines the gradient of log-likelihood as a score function and optimizes them by the theory of denoising score matching. The sampling process of the score-based model includes ODE, Langevin, and Predictor-Corrector [31], and they have different sampling efficiency and quality.

Recently, some other work has been designed to combine the benefit of different forms of DGMs. Reference [50] defines that when the optimal discriminator in GAN is achieved, the generator can be treated as an EBM. CooperNet [62] and VAEBM [63] combine the advantages of VAEs and EBMs, and [64] learns EBM with recovery likelihood like DDPM. Sliced score matching [30] is also widely used in EBMs.

Recently, [65] extend the usage of DGM as a data source for downstream tasks, and they discover that a well trained generative model can provide a good source of data for downstream tasks like classification. In this paper, we mainly explore the effect of the DGM on KD, and analyze how to provide a proper DGM for KD.

III. DGM FOR KD

A. BASIC SETTINGS

Knowledge Distillation(KD) is formulated by a generative distribution, and it optimizes the following objective function:

$$\min_{\phi_s} \mathcal{L}_{kd} = \mathbb{E}_{x\sim p_\theta(x)}[\mathrm{KL}(p_{\phi_t}(z|x)||p_{\phi_s}(z|x))]. \tag{1}$$

The distribution of output logit $p_\phi(z|x)$ typically is some widely-used discrimination architectures, like ResNet and VGG. Moreover, in the full-data KD problem, the sampling process $x \sim p_\theta(x)$ is from the data distribution p_{data}. When p_{data} is not available, it falls into DFKD, and it tends to sample from the marginal distribution $p_\theta(x) = \int p_\theta(x|z)p(z)\mathrm{d}z = \mathbb{E}_{z\sim p(z)}[p_\theta(x|z)]$. Therefore, by using a reparameterize trick, we model distribution $p_\theta(x|z)$ by a generator. Then the optimization of DFKD can be concluded as a likelihood

TABLE 1. Previous DGMs, as well as their training and sampling strategy.

Method	Training Objective	Sampling Process	Sampling Cost	$p(z)$
GAN [17], [66]	equation 2	$x = G(z)$	$O(1)$	$\mathcal{N}(0, \mathbf{I})$
VAE [19]	equation 4	$x = G(z)$	$O(1)$	$\mathcal{N}(\mu_\theta(x), \sigma^2_\theta(x))$
Flow [19], [20]	$\ln q_K(\mathbf{z}_K) = \ln q_0(\mathbf{z}_0) - \sum_{k=1}^{K} \ln \left\| \det \frac{\partial f_k}{\partial \mathbf{z}_{k-1}} \right\|$	$x = \mathbf{z}_K = f_K \circ \ldots \circ f_2 \circ f_1(\mathbf{z}_0)$	$O(1)$	for $z_0, \mathcal{N}(0, \mathbf{I})$
EBM [50], [67]	$\frac{\partial \log p_{\boldsymbol{\theta}}(\boldsymbol{x})}{\partial \boldsymbol{\theta}} = \mathbb{E}_{p_{\boldsymbol{\theta}}(\boldsymbol{x}')} \frac{\partial E_{\boldsymbol{\theta}}(\boldsymbol{x}')}{\partial \boldsymbol{\theta}} - \frac{\partial E_{\boldsymbol{\theta}}(\boldsymbol{x})}{\partial \boldsymbol{\theta}}$	$x^-_{t+1} = x^-_t + \frac{\sigma^2_t}{2} \frac{\partial f_\theta(x^-_t)}{\partial x^-_t} + \epsilon_t$	$O(T)$	None.
DDPM [25], [27]	$L_{\text{vlb}} := L_0 + L_1 + \ldots + L_{T-1} + L_T$ $L_0 := -\log p_\theta(x_0 \mid x_1)$ $L_{t-1} := D_{KL}(q(x_{t-1}) \| p_\theta(x_{t-1}))$ $L_T := D_{KL}(q(x_T \mid x_0) \| p(x_T))$	$x_T \sim \mathcal{N}(0, \mathbf{I})$ $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}(\mathbf{x}_t - \beta_t \epsilon_\theta) + \sigma_t \mathbf{z}$	$O(T)$	None.
SDE [31]	$\mathbb{E}_t\{\lambda(t)\mathbb{E}_{\mathbf{x}_0}\mathbb{E}_{\mathbf{x}_t \mid \mathbf{x}_0} [\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t) - \nabla_{\mathbf{x}_t} \log p_{0t}(\mathbf{x}_t)\|_2^2]\}$	Langevin: Same as EBM PC: SGLM predictor + Langevin Corrector ODE solver	Langevin: $O(T)$ PC: $O(T)$ ODE: t_{ode}	Probability Flow.

estimation problem, i.e.,

$$\min_\theta \mathbb{E}_{x \sim p_\theta(x|z), z \sim p(z)}[\mathcal{F}(p_{\phi_t}(z|x), p_{\phi_s}(z|x))].$$

DFKD methods focus on the design of different $\mathcal{F}$s. Different components are implemented as follows:

- Latent code prior $p(z)$, typically $\mathcal{N}(0, \mathbf{I})$. Some work finetunes z for a more balanced class generation.
- Decode process $p_\theta(x|z)$, typically a generator as a GAN does.
- Likelihood function $\mathcal{F}(\ldots)$typically constraints the quality of generation, including class balancing, adversarial training, difficulty measuring, or constructive samples. Previous DFKD methods focus on the design of the likelihood function.

TABLE 2. Performance of KD by different DGMs. All results are performed at CIFAR10.

Teacher	Student	G Method	Top1-Acc	IS	FID
ResNet34 (Teacher Acc: 95.70%)	ResNet18 (Student Acc: 95.20%)	Vanilla KD	93.64	–	–
		DCGAN	94.50 ± 0.01	7.13	180.3
		DDPM	$\mathbf{94.91 \pm 0.01}$	7.97	11.2
		DDIM	94.73 ± 0.03	7.94	11.1
		EBM	93.83 ± 0.03	7.32	67.8
		SDE(ODE)	82.05 ± 0.33	9.33	**3.00**
		SDE(PC)	72.35 ± 0.23	**9.85**	4.57
VGG11 (Teacher Acc: 92.20%)	ResNet18 (Student Acc: 95.20%)	Vanilla KD	89.41	–	–
		DCGAN	90.92 ± 0.02	7.13	180.3
		DDPM	$\mathbf{92.07 \pm 0.03}$	7.97	11.2
		DDIM	92.02 ± 0.05	7.94	11.1
		EBM	91.51 ± 0.08	7.32	67.8
		SDE(ODE)	74.25 ± 0.36	9.33	**3.00**
		SDE(PC)	63.23 ± 0.34	**9.85**	4.57

B. PRELIMINARY ON DGMs

In this section, we explore the design of p_θ. As it related to the distribution of the data space, we can use DGMs. We give some formulations of recent DGMs.

Generative Adversarial Networks(GANs) [16], [17], [68]. They are widely-used generative models to get high-quality images, with two deep networks. Specifically, a generator $G$to model the generative distribution $p(x|z)$, and a discriminator D to discriminate whether a data sample is real or fake. The training of GAN is a min-max game, i.e.,

$$\min_G \max_D F(G, D) = \mathbb{E}_{x \sim p_{data}}[\log D(x)] + \mathbb{E}_{z \sim p(z)}[\log(1 - D(G(z)))], \quad (2)$$

When the optimial discriminator $D^*$is learned (by setting $\frac{\partial F(G,D)}{\partial D} = 0$), the objective function above is to learn $\min_G \text{JS}(p_{data}(x)||p_g(x)) + 2\log 2$. There are many variants to stabilize the training of GAN, including DCGAN [68], WGAN [16], and StyleGAN [69]. We use pretrained DCGAN to generate pseudo samples for KD, which is also widely used in the design of generators in many DFKD tasks.

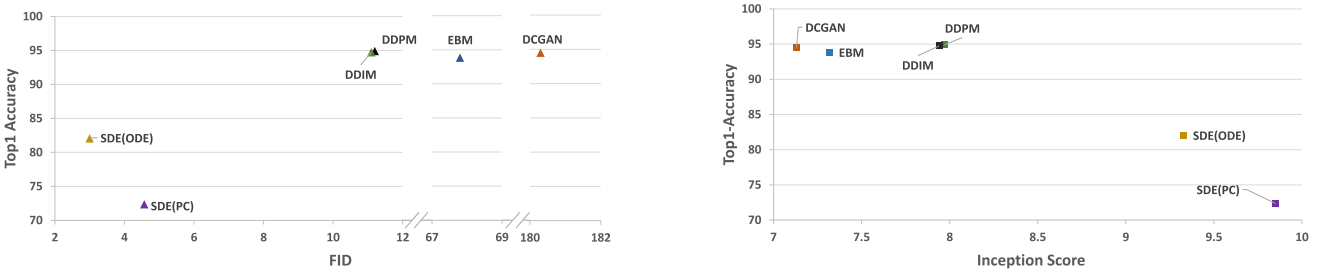
Variational autoencoders(VAEs) [53], [70] Though GAN achieves great success in generation tasks, the training process of GAN still lacks training stability and thus easily causes mode collapse. Thus some likelihood estimation methods are proposed to improve the training stability and sample diversity. The maximization likelihood estimation is formulated as

$$\max_\theta \mathbb{E}_{x \sim p_{data}}[\log p_\theta(x)]. \quad (3)$$

One of the important likelihood estimation ideas is the variational autoencoder. They optimize Evidence Lower Bound (ELBO), i.e.

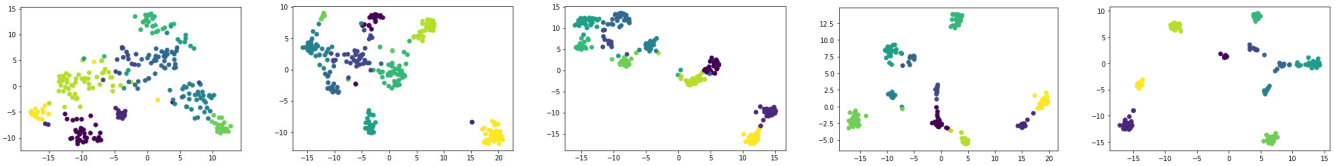
$$\log p_\theta \geq \mathbb{E}_{z \sim q_\phi}\left[\log p_\theta(x \mid z)\right] - D_{KL}\left(q_\phi(z \mid x) \| p_\theta(z)\right) = \mathcal{L}_{ELBO}. \quad (4)$$

MLE-based methods. In the context of VAEs, the maximum likelihood estimation (MLE) technique, represented by equation 3, is commonly used to optimize the parameters of the data distribution. However, alternative methods have emerged recently. **The training process** for VAEs involves learning the parameters of θ, followed by obtaining pseudo data samples from p_θ, which is referred to as **the generation process**. A summary of the related deep generative models (DGMs) is provided in Table 1.



(a) Related with FID. (b) Related with IS.

FIGURE 2. The performance of KD by pseudo samples from different pretrained DGMs at different measures of sample quality. We use IS(a) and FID(b) to measure image quality. Better viewed on screen.



(a) GANs. (b) EBM. (c) DDPM. (d) SDE(ODE). (e) SDE(PC).

FIGURE 3. t-SNE visualization of the feature map of the last layer of teacher model. The color represents the pseudo label of each pseudo sample, calculated by $y_i = \arg\max_i z_{t,i}$. Better viewed on screen.

MLE methods offer a higher theoretical log-likelihood for the data distribution and are usually optimized using the Langevin dynamic sampling strategy. The gradient of log-likelihood is included in all sample update steps to measure the distance between the data distribution and the learned distribution. Table 1 summarizes the related DGMs, which first learn the parameter of θ in the training process and then obtain pseudo data samples from p_θ.

C. PERFORMANCE FOR KD BY DIFFERENT DGMs

In this section, we explore the potential of using pseudo samples generated by deep generative models (DGMs) for knowledge distillation (KD) in image generation tasks. Specifically, we conduct experimental analysis on the performance of KD models trained with pseudo samples obtained from different data distributions p_θ. The DGMs used in our experiments include GAN [17], DDPM [25], [26] and DDIM [29], JEM++ [67], and score-based models with SDE and different sampling strategies, such as predictor-corrector (PC) and ordinary differential equation (ODE) [31]. To obtain the pseudo samples, we use the pretrained weights of the DGMs and generate samples from the learned p_θ. The generated samples are then used to train the student models.

The experimental results, presented in Table 2, demonstrate a substantial enhancement in the performance of KD by using pseudo samples from the pretrained generative model to guide the data distribution. However, it is noteworthy that the pretrained SDE model did not perform as well, which will be discussed in section III-D.

To evaluate the generalization capability of the generated data, we conducted the same experiments with a different architecture for the teacher model, specifically in VGG11 [71], and obtained similar results. Moreover, we observed better improvements in the vanilla KD structure. These findings suggest that utilizing different distillation architectures is crucial for enhancing the quality of the generated data.

D. RELATIONSHIP BETWEEN IMAGE QUALITY AND KD PERFORMANCE

Based on the results in Table 2, we conclude that the quality of the generation stage has a positive impact on the performance of DFKD, except for the score-based model. To investigate the reason for this and further improve the performance of DFKD, we need to examine the distribution of the pretrained generative models and assess the influence of the data distribution on the performance of KD. In this regard, we use two commonly used metrics, namely, inception score (IS) [72] and Fréchet Inception Distance (FID) [73], to measure the quality of the sampled images from the pretrained DGMs.

The results are presented in Figure 2, which shows that, although the performance of KD does not increase significantly with the higher quality of pseudo samples, the performance of SDE is significantly lower than that of other models, despite having high IS and low FID scores. To investigate this further, we perform t-SNE visualizations of the output features of the teacher model for different DGMs in CIFAR10. We sample 256 images for each method and assign different pseudo labels to them, using different colors for visualization.

From the visualization results in Figure 3, we observe that the pretrained GAN model can greatly improve the performance of KD. Moreover, the t-SNE visualization of the score-based model shows that the learned distribution of images is highly discriminative among different classes but lacks

diversity within each class, leading to poor performance. As described in adversarial-based DFKD methods and CMI [14], data diversity is crucial for the KD task, as it can mine more challenging samples and improve the robustness of KD. Therefore, to better utilize DGMs for training KD, we suggest adding the probability flow of adversarial samples to DGMs.

IV. SAMPLING WITH PRIOR

In the previous section, we explore how different designs of p_θ can affect the output feature map of the teacher model and the performance of generative KD. However, this information only offers a limited understanding of the original data distribution and how well the generative KD performs. In this section, we propose a more detailed framework called GenKD, which is inspired by the main idea of DFKD. To extract knowledge from the KD framework, we aim to estimate the gradient flow from $\nabla_{x^-}\mathcal{L}_{kd}$ to $\nabla_\theta x^-$. Figure 4 illustrates the training and generation stages of our GenKD framework, which involve the following steps: (1) training the EBM module using maximum likelihood estimation (MLE) to learn the energy distribution p_θ, as well as leveraging the knowledge from the pretrained teacher network and pseudo data buffer; and (2) training the KD model using the pretrained energy module and pseudo data buffer [7].

A. LOSS FUNCTION FOR IMAGE QUALITY

To extract knowledge from the teacher model and KD framework, as previous work on DFKD has done [8], [9], [10], [14], it is crucial to carefully design the downstream loss function $\mathcal{L}_c$ for the KD task.

Adversarial DFKD methods show that generating samples that are useful for training KD is essential, even though it goes against the objective of traditional KD. This adversarial approach can assist EBM in generating challenging samples for both the teacher and student models.

$$\max_\theta \mathcal{L}_{kd} = \mathbb{E}_{x^-\sim p_\theta}[\mathrm{KL}(p_{\phi_t}(y^-|x^-)||(p_{\phi_s}(y^-|x^-))], \quad (5)$$

Equation 6 can ensure that the positive and negative samples exhibit similar behavior in downstream tasks. Additionally, the distribution of the teacher model on generated data x^- and real data x^+ should be sufficiently close. This requirement can be expressed mathematically as follows:

$$\min_\theta \mathcal{L}_{kl} = \mathbb{E}_{x^-\sim q_\theta}\mathrm{KL}(p_{\phi_t}(y^-|x^-)||p_{\phi_t}(y^+|x^+)). \quad (6)$$

Moreover, the generated samples from p_θ should be correctly classified by pretrained teacher model $p_{\phi_t}(y|x^-)$, i.e.,

$$\mathcal{L}_{cls} = \mathbb{E}_{x^-\sim p_\theta}[-\sum_{i=1}^{C}\hat{y}_i \log(p_{\phi_t}(y_i|x^-)]. \quad (7)$$

Therefore, the total downstream loss $\mathcal{L}_c$ can be represented as

$$\mathcal{L}_c = \lambda_{kl}(\mathcal{L}_{kl} - \mathcal{L}_{kd}) + \lambda_{cls}\mathcal{L}_{cls}. \quad (8)$$

The loss function $\mathcal{L}_c$ can be interpreted as the negative log-likelihood of a distribution p_c that is related to the data distribution. In our approach, we define the condition c as a random variable that describes the regularization of the training stage in DFKD. This condition is related to the status of both the teacher model and the student model. To sample data from a given DGM, we use the gradient of the log-likelihood, which is defined as follows:

$$\begin{aligned}\nabla_x \log p_\theta(x, c) &= \nabla_x \log p_\theta(x|c)p(c) \\ &= \nabla_x(\log p_\theta(x|c) + \log p(c)) \\ &= \nabla_x(\log p_\theta(x|c) - \mathcal{L}_c). \end{aligned} \quad (9)$$

The likelihood $p_\theta(x|c)$ is obtained from a pretrained DGM. As shown in Table 2, despite the availability of a large number of pseudo labels, EBMs fail to significantly improve the performance of KD. Therefore, we consider using EBMs as an example to demonstrate how to fine-tune the DGM with the downstream probability flow $p(c)$. Since obtaining an accurate maximum likelihood with EBM is challenging, we focus on finetuning the DGM instead.

B. CONVERGENT MCMC SAMPLING FOR MLE

At the training stage, the purpose of step (ii) in figure 4 is to extract the knowledge from teacher models, i.e., we need to optimize EBMs. The MLE of EBM is to optimize,

$$\max_\theta \mathcal{L}(\theta) = \mathbb{E}_{x\sim p_{data}}[\log p_\theta(x)]. \quad (10)$$

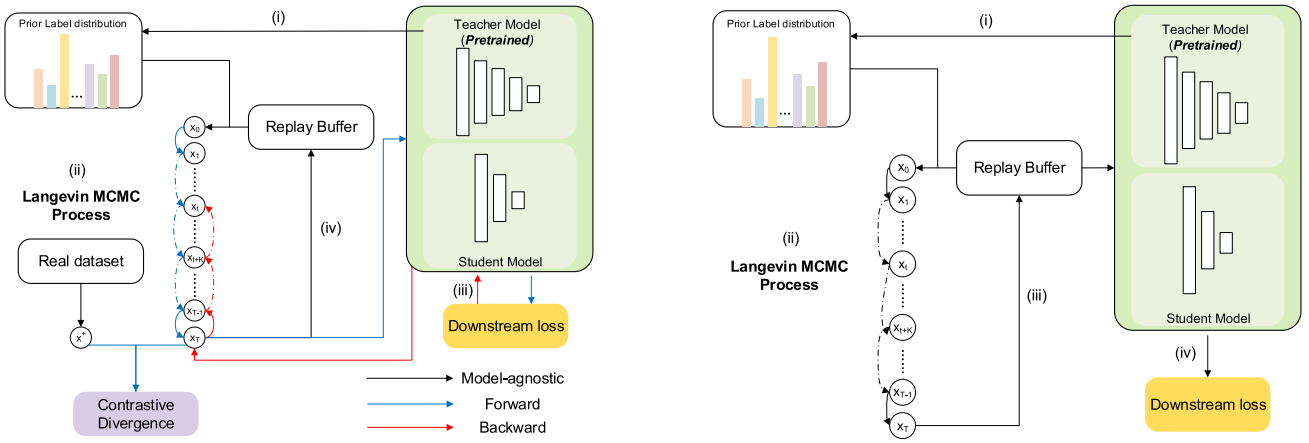
Equation 10 equals minimizing the KL divergence between generation distribution $p_\theta(x)$ and real distribution $p_{data}(x)$. Therefore, to minimize the negative log-likelihood of EBM, the gradient is

$$\frac{\partial \mathcal{L}(\theta)}{\partial \theta} = \mathbb{E}_{x^+\sim p_{data}}[\frac{\partial f_\theta(x)}{\partial \theta}] - \mathbb{E}_{x^-\sim p_\theta}[\frac{\partial f_\theta(x)}{\partial \theta}]. \quad (11)$$

$f_\theta(x)$ represents the negative energy function. When using label information, it can be reinterpreted as $f_\theta(x, y)$. However, the second item of equation 11 is intractable. Thus, the sampling process $\mathbf{x}^- \sim p_\theta$ can be approximated by MCMC methods like Langevin dynamics [74]. At a specific time step t, the update step is

$$x_{t+1}^- = x_t^- + \frac{\sigma_t^2}{2}\frac{\partial f_\theta(x_t^-)}{\partial x_t^-} + \epsilon_t. \quad (12)$$

The noise vector $\epsilon_t \sim \mathcal{N}(0, \sigma_t^2\mathbf{I})$ and the t-step sampling instance $x_t^- \sim q_\theta^t$, where q_θ^t is the distribution obtained after t steps of MCMC sampling. If t approaches infinity and σ_t approaches zero, q_θ approaches the pretrained DGM p_θ. However, due to the presence of local modes, the sampling can result in divergence during the EBM training process. To overcome this issue, short-run MCMC was proposed by [23] for non-convergent non-persistent EBMs. They approximate the data distribution p_{data} instead of maximum likelihood estimation (MLE). In GenKD, we adopt convergent EBMs to preserve the MLE for downstream KD tasks.



(a) Training Stage (b) Generation Stage

FIGURE 4. The diagram of GenKD. (a) The training stage of GenKD. (i) Generate label prior distribution $p(y)$ by pretrained teacher model. (ii) Langevin MCMC process from replay buffer by equation 12, and compute downstream losses. (iii) Compute and backward the gradient from KD framework. (iv) Update the EBM and replay buffer, repeat step (ii) until convergence. (b) The generation stage of GenKD. (i) (ii) are the same as (a). (iii) Update replay buffer. (iv) Train a KD framework by $\min_{\phi_S} \mathrm{KL}(p_{\phi_t}(y|x)||p_{\phi_S}(y|x))$. Better viewed on screen.

C. POLICY GRADIENT FOR UPDATING θ

Now we update the EBM module by the extracted knowledge from $\mathcal{L}_c$. The learning objective is

$$\min_{\theta} \mathbb{E}_{x^- \sim p_\theta}[\mathcal{L}_c(x)]. \tag{13}$$

The gradient of equation 13 is calculated as

$$\nabla_\theta \mathbb{E}_{x^- \sim p_\theta}[\mathcal{L}_c(x^-)] = \mathbb{E}_{x^- \sim p_\theta}[\mathcal{L}_c(x) \nabla_\theta \log p_\theta(x)]. \tag{14}$$

However, the sampling process of $\mathbf{x}^- \sim p_\theta$ is non-differentiable and non-derivative. To address this issue, we adopt policy gradient techniques [75], [76] from reinforcement learning. Specifically, based on Equations 13 and 14, we consider the update process of θ as choosing the best paths to decrease $\mathcal{L}_c$ and gradually approach the MLE of EBM. We propose a policy gradient algorithm to update GenKD, and the details are provided in Section V-B.

Since the Langevin MCMC sampling strategy is a Markov Chain, we break down the sampling process into several single-step Langevin processes. By leveraging the Markov property, we can approximate the expected gradient as follows:

$$\nabla_\theta \mathbb{E}_{x^- \sim p_\theta}[\mathcal{L}_c(x^-)] = \sum_{t=0}^{T-1} \mathcal{L}_c(x_t) \nabla_\theta p_\theta(x_{t+1}|x_t) \tag{15}$$

Here $T = \infty$ if using convergent persistent MCMC process to update, and $T \in \{80, 100\}$ in short-run MCMC by settings of previous work [23]. In practice, we set $T = \infty$ by the design of replay buffer in many previous work [22], [50] for building a convergent EBM.

D. K-STEP LANGEVIN MCMC POLICY GRADIENT FOR MLE

Specifically, we define the policy π_θ in terms of the K-step Langevin MCMC transition probability $M_\theta(x^-_{t+K}|x^-_t)$, which is similar to the definition of policy in reinforcement learning. To evaluate the performance of the policy, we can use

Algorithm 1 Training Stage of GenKD

Require: Training dataset $\mathcal{T}_{tr}$, Pretrained teacher model $p_{\phi_t}(y|x)$, step size σ, noise coefficient ϵ, Replay probability μ.

Ensure: Generated dataset $\mathcal{T}_{gen}$

Initialize $\mathcal{B}$ with $\mathcal{U}(-1, 1)$.
$w_t \leftarrow$ Weight of last layer in ϕ_t
$\alpha \leftarrow f(w_t)$
repeat
  Sample training batches $x^+ \sim \mathcal{T}_{tr}, y \sim Dir(\alpha)$
  $x_0^- \sim \mathcal{B}$ with probability μ otherwise $\mathcal{U}(-1, 1)$
  $s = []$.
  for $i = 0$ **to** $T - 1$ **do**
    $\sigma_k \sim \mathcal{N}(0, \sigma_k^2 \mathbf{I})$
    $x_{k+1}^- = x_k^- + \frac{\sigma_k^2}{2} \frac{\partial f_\theta(x_k^-, y)}{\partial x_k^-} + \epsilon_k$
    s.append$((x_{k+1}^-, x_k^-))$
  end for
  $x^- = x_T^-$.detach()
  $L_a = \frac{1}{b} \sum_{j=1}^{b} (f_\theta(x^+, y) - f(x^-, y))$
  {*Sample K-steps for policy gradient* }
  $t \sim \{0, 1, \ldots, T-K-1\}$
  for $i = t$ **to** $t + K$ **do**
    $\mathcal{L}_c$ calculated by equation 8.
    $L_b = \mathrm{SinglePG}(t, \mathcal{L}_c, s)$
  end for
  $L_{cls} = CE(p_\theta(\hat{y}|x), y)$
  $L = L_a + \lambda_a L_b + \lambda_{cls} L_{cls}$
  L.backward()
until EBM is converged

Algorithm 2 Single Policy Gradient step(Single PG)

Require: time step t, downstream loss $\mathcal{L}_c$, sampling list s
Ensure: Reward increment L_b
$x_{t+1}^-, x_t^- = s[t]$
set Gaussian distribution $q_\theta = \mathcal{N}(x_t^- + \frac{\sigma_t^2}{2} s_\theta(x_t^-), \sigma_t^2)$
$x_{t+1}^- \sim M_\theta(x_{t+1}^- | x_t^-) = \mathcal{N}(x_t^- + \frac{\sigma_t^2}{2} s_\theta(x_t^-), \sigma_t^2)$
Compute gradient $\frac{\partial l(\theta)}{\partial \theta}$ by equation 18
$L_b = r(x_i^-, x_{i+1}^-) \log M_\theta(x_{i+1}^- | x_i^-)$

the reward function defined as the decrease in the value of $\mathcal{L}c$, i.e., $r(x_t^-, xt + K^-) = \triangle\mathcal{L}_c(x_t^-, x_{t+K}^-) = \mathcal{L}_c(x_t^-) - \mathcal{L}_c(x_{t+K}^-)$. Therefore, we need to find the optimal policy parameters θ that maximize the expected reward. However, directly optimizing the objective in Equation 9 can be computationally expensive in deep learning, as it involves $O(T)$ operations at each forward and backward pass. To mitigate this issue, we can sample K steps of Langevin MCMC to estimate the expected reward, as the Langevin MCMC sampling process has a Markov characteristic. The problem can be defined as,

$$\max_\theta \mathbb{E}_{x_{t+K}^- \sim M_\theta(x_{t+K}^- | x_t^-)}[r(x_t^-, x_{t+K}^-)]. \tag{16}$$

We can implement a policy gradient algorithm to update θ. When σ_t is small, the single-step Langevin MCMC process is a Markov Process, i.e. $x_{t+1}^- \sim M_\theta(x_{t+1}^- | x_t^-) = \mathcal{N}(x_t^- + \frac{\sigma_t^2}{2} s_\theta(x_t^-), \sigma_t^2)$. Therefore, if $M_\theta(x_{i+1}^- | x_i^-) = M_{\theta,1}$, the final tractable gradient can be derived as,

$$\max_\theta l(\theta) = \sum_{i=t}^{t+K-1} \mathbb{E}_{x_{i+1}^- \sim M_{\theta,1}}[r(x_i^-, x_{i+1}^-)], \tag{17}$$

$$\nabla_\theta l(\theta) = \sum_{i=t}^{t+K-1} \mathbb{E}_{x_{i+1}^- \sim M_{\theta,1}}[r(x_i^-, x_{i+1}^-) \nabla_\theta \log M_{\theta,1}]. \tag{18}$$

Here the gradient item can be directly calculated due to the Gaussian distribution of $M_\theta(x_{i+1}^- | x_i^-)$. The detailed algorithm specification can be referred to in Algorithm 1 and 2.

V. DISCUSSION ON GenKD

Algorithm 1 samples continuous T steps of the Langevin MCMC update process for reward calculation and policy gradient. This process is essential for accurately estimating the expected reward and updating the policy. In this section, we present a brief theoretical analysis of GenKD, which aims to provide insights into the properties of the algorithm and its performance. This analysis is important for understanding the strengths and limitations of GenKD and for guiding future improvements.

A. LIKELIHOOD ESTIMATION

To improve the knowledge distillation process for energy-based models (EBMs), it is important to understand the effect of downstream loss $\mathcal{L}_c$ on the MLE estimation of the EBM. Previous DFKD methods have attempted to optimize the input data $\mathbf{x}$ directly, which can lead to getting stuck in local modes and decrease diversity in the generation stage, harming the KD task.

Equation 10 presents the MLE estimation of the EBM distribution p_θ, which serves as the foundation for knowledge distillation. The following theorem clarifies that when optimizing the downstream loss $\mathcal{L}c$ along with the negative log-likelihood $\mathbb{E}_{x \in p_{data}}[-\log p_\theta(x)]$, the learned EBM distribution p_θ will be sufficiently close to the true data distribution p_{data}. This insight can guide the development of more effective and accurate knowledge distillation methods for EBMs.

B. CONNECTION TO POLICY GRADIENT

The policy gradient algorithm [76] is a commonly employed technique for function approximation in the field of reinforcement learning [75]. This method involves parameterizing the policy by θ and updating the value function based on the gradient of the expected reward with respect to the policy parameters. For any MDP, the long-term expected reward per-step $\rho(\pi_\theta)$ is

$$\frac{\partial \rho}{\partial \theta} = \sum_s d^\pi(s) \sum_a \frac{\partial \pi(s,a)}{\partial \theta} Q^\pi(s,a)$$

The policy is modeled by deep networks [77], so by function approximation, the parameter of the policy, θ can be updated with gradients of an approximated function $f_\pi(s,a) = Q^\pi(s,a)$.

Considering GenKD, we can correlate the MDP in Langevin dynamics with MDP in reinforcement learning. In this analysis framework, we assume to run ∞ steps of Langevin dynamics(equation 12). The framework can be,

- **State** s**.** The generated sample from specific time step t at Langevin dynamics.
- **Action** a**.** One single step dynamic x^t to x^{t+1}.
- **Policy** π_θ. The distribution of single step Langevin dynamic $M_\theta(x_{t+1}^- | x_t^-)$.
- **Approximation function of value function** f_w**.** The negative downstream loss $-\mathcal{L}_c$.
- **Stationary distribution** $d^\pi(s)$**.** Here we set d_s^π as uniform distribution because at each time step the distribution of generated data is close enough.

As explained in the related literature on policy gradient algorithms, our GenKD model can be considered a specific framework for function approximation within this class of methods, allowing for the optimal determination of θ. Theorem 1 demonstrates that our GenKD framework can theoretically enhance the performance of DFKD and enable more efficient search within a smaller data space.

VI. EXPERIMENTS ON GenKD

In this section, we evaluate the performance of the GenKD model by examining two aspects: 1) its ability to generate related image samples to improve the performance of KD,

TABLE 3. Performance of generation and DFKD compared different methods on CIFAR10, CIFAR100, and SVHN. N: size of pseudo samples, and t_{inv}: GPU hours for the generation(on 3090TI). GenKD(N): training GenKD without policy gradient, and we set $K = 1$ in this table.

	CIFAR10				CIFAR100				SVHN			
Method	N	IS	t_{inv}	S. Acc	N	IS	t_{inv}	S. Acc	N	IS	t_{inv}	S. Acc
noise	50.0k	1.25	-	10.00	50.0k	1.25	-	1.01	50.0k	1.23	-	10.00
DAFL [12]	51.2k	2.31	3.79	55.40	76.8k	2.06	5.71	36.31	38.4k	2.53	3.68	92.80
DeepInv [10]	128k	2.91	17.8	52.13	179k	4.20	23.7	31.29	96.0k	2.49	12.4	36.02
CMI [14]	51.2k	1.63	23.7	40.94	76.8k	4.95	33.5	61.55	38.4k	1.53	19.2	40.49
GenKD(N)	50.0k	**7.67**	**3.25**	68.77	50.0k	**7.01**	**6.50**	47.55	50.0k	3.33	**2.75**	94.21
GenKD	50.0k	7.23	**3.25**	70.19	50.0k	6.86	**6.50**	50.13	50.0k	2.88	**2.75**	95.35
GenKD	100.0k	7.36	4.70	**72.52**	100.0k	6.84	9.40	**61.87**	100.0k	3.26	4.00	**95.66**

and 2) its ability to improve the generalization of existing KD methods. To generate the related image samples, we implement JEM [50] and JEMPP [67] for generating conditional pseudo samples. These models can model both generative models $p_\theta(x)$ or $p_\theta(x|y)$, and discriminative models $p_\theta(y|x)$, which can benefit the downstream models. However, a minor difference between our EBM and JEM is that we construct the conditional distribution $p_\theta(x|y)$ as EBM instead of the joint distribution $p(x, y)$.

A. DATASETS AND IMPLEMENTATION DETAILS

We implement our GenKD model on different datasets, and our implementation is based on Pytorch [78]. The detailed description of each dataset is:

- **CIFAR10** and **CIFAR100** [79]. A dataset for the classification of some scene images with resolution $32 \times 32 \times 3$.
- **SVHN** [80]. Real-world image dataset with house number.

In contrast to JEM, we model the conditional distribution $p(x|y)$ instead of the joint distribution $p(x, y)$ for the negative energy function $f_\theta(x, y)$, i.e., $p_\theta(x|y_i) = \exp(f_\theta(x, y))$. The posterior distribution for the classification model $p_\theta(y|x)$ can be expressed as:

$$p_\theta(y_i|x) = \frac{p_\theta(x, y_i)}{p_\theta(x)} = \frac{p_\theta(x|y_i)p(y_i)}{\sum_{j=1}^{C} p_\theta(x|y_j)p(y_j)}.$$

Thus, compared with the negative energy function $f(x, y)$, the logit before the *softmax* operation is $l_i = f_\theta(x, y_i) + \log p(y_i)$. To model the prior distribution of label $p(y)$, we use the Dirichlet distribution, motivated by [7]. We use the framework of JEMPP [67] and set SGLD with noise $\sigma = 0.02$ for all benchmarks.

B. PERFORMANCE ON KD

To evaluate the performance of KD, we first need to determine whether our re-implemented JEM method is valid for the downstream KD task. To do this, we compare the inversion capability of DAFL [12], DeepInversion [10] and CMI [14]. We also implement the KD framework on noise data to provide a baseline for the generation. For the performance of DFKD, we set the teacher model as resnet32 $\times$ 4 [81] and the student model as resnet8 $\times$ 4 [81]. We use trained replay buffers as pseudo samples of DFKD and validate the KD model by the original dataset. All comparison methods use hyperparameters provided by their respective papers.

We present the quantitative performance of GenKD against other DFKD frameworks in Table 3. Our comparison mainly focuses on three aspects: 1) generation performance measured by IS and FID, 2) DFKD performance evaluated by student accuracy, and 3) generation time reflecting the time cost of the training stage. As shown in Table 3, our GenKD outperforms other DFKD methods in terms of the quality of pseudo samples and the required data size for effective search, with less time consumption. Our method also generates high-quality pseudo samples that are visually appealing compared to other DFKD approaches. Unlike DFKD, our GenKD is equipped with data prior and is therefore more focused on image quality. Incorporating $\mathcal{L}_c$ to the model results in some sacrifice in image quality, but it ensures the superior performance of KD by finding pseudo samples that are more beneficial to the training, as opposed to simply generating better images. The result agrees with our discovery in section III.

In addition, we extend our GenKD model to all datasets and visualize the generated replay buffer in Figure 5. The visualization illustrates different instances of samples with different labels and high-quality pseudo samples, with diversity generation performance.

C. ABLATION STUDY AND HYPERPARAMETER SENSITIVITY

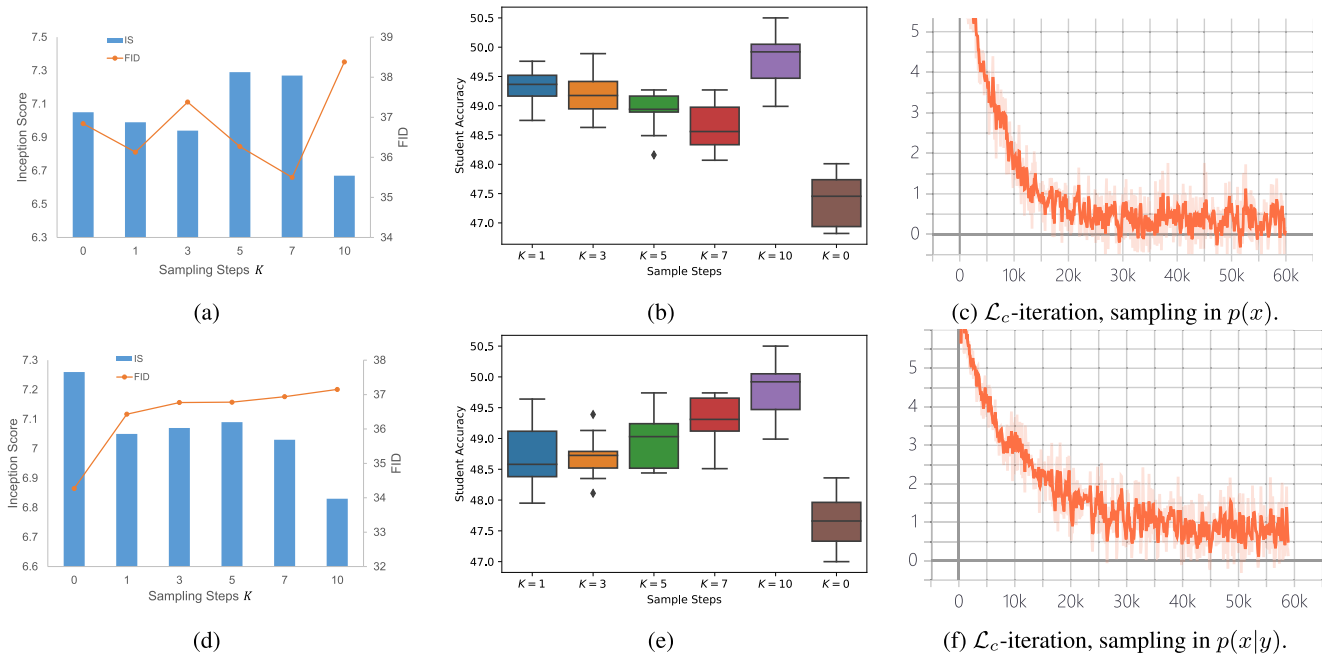
In this subsection, we explore the performance of generation and KD with different modules and hyperparameters. Thus, we check the following factors in our GenKD module:

- **The sampling methods**: sampling from $p_\theta(x)$ or $p_\theta(x|y)$.The label information can be very important in training KD.
- **The effect of $\mathcal{L}_c$ policy gradient item on the performance of KD.** We need to know if such a plug-in module is valid.
- **The sampled steps K for policy gradient.** The length of the Markov chain, we assume it's related to how we approach the solution of equation 8.

In addition, various factors such as the starting point of the policy gradient t, detailed hyperparameters such as λ_{kd} and λ_{cls} for the downstream loss, and hyperparameters of the EBM, such as the step size and data noise, can also play a critical role. In this section, we specifically examine the

(a) Visualization of pseudo CIFAR100. (b) Visualization of generated SVHN. (c) Visualization of generated CIFAR10.

FIGURE 5. The generation/inversion results of our generation stage sampled by the distribution $p(x|y)$. We sample one image for each class in dataset CIFAR100, and 10 images for each class in dataset SVHN and CIFAR10. Better viewed on screen.



(a) (b) (c) $\mathcal{L}_c$-iteration, sampling in $p(x)$.

(d) (e) (f) $\mathcal{L}_c$-iteration, sampling in $p(x|y)$.

FIGURE 6. The quantitative performance of generation (a), (d) and KD (b), (e) sampled from different distributions. (a-c) are from $p(x)$, and (d-f) are from $p(x|y)$. The convergence curves of (c) and (f) are plotted with $K = 10$. Better viewed on screen.

impact of the policy gradient on the generation performance and the DFKD results.

Sampling strategies. We use negative energy function $f(x)$ or $f(x, y)$ to represent the sample process from $p(x)$ or $p(x|y)$ in our implementation. When training the EBM, we calculate the contrastive divergence L_a for different sampling strategies, following the approach in JEM [50], [67]. Table 4 shows the quantitative results.

We compare the generation and DFKD performance and conduct our ablation study experiments on CIFAR100. For the generation results, when we sample from the marginal distribution $p_\theta(x)$, the generation performance can be slightly better with about 0.13 IS and 0.5 FID score. This improvement is because $p_\theta(x) = \sum_{i=1}^{C} p_\theta(x|y)p(y)$ aggregates the information of different classes, and the visual information learned by the distribution can increase the diversity among different classes.

Effect of $\mathcal{L}_c$ on KD performance. In Table 4, we present quantitative results on the effect of adding downstream loss $\mathcal{L}_c$ for downstream tasks. In the table, we set $\mathcal{L}_c$ to 'N' when we do not implement lines 15 to 20 in Algorithm 1. We observe that as the number of sampling steps K increases, the generation quality initially improves, then reaches a peak and begins to decline. Simultaneously, the performance of

TABLE 4. The effect of sampling steps K for policy gradient. During KD, the teacher model is resnet32 × 4, and the student model is resnet8 × 4. The top-1 accuracy on the teacher model is 72.09%. The size of training pseudo samples is 40000.

$p(x)$	$p(x\|y)$	$\mathcal{L}_c$	K	IS	FID	Stu Acc.
Y	N	N	_	7.05	36.84	47.30 ± 0.23
Y	N	Y	1	6.99	36.13	49.33 ± 0.29
Y	N	Y	3	6.94	37.38	49.46 ± 0.31
Y	N	Y	5	7.29	36.27	48.91 ± 0.33
Y	N	Y	7	7.27	35.50	48.62 ± 0.42
Y	N	Y	10	6.67	38.38	**49.82 ± 0.43**
N	Y	N	_	7.26	**34.27**	47.64 ± 0.40
N	Y	Y	1	7.05	36.43	48.71 ± 0.51
N	Y	Y	3	7.07	36.77	48.72 ± 0.35
N	Y	Y	5	7.09	36.78	48.96 ± 0.43
N	Y	Y	7	7.03	**36.94**	49.30 ± 0.38
N	Y	Y	10	6.83	37.15	49.56 ± 0.28

KD also improves, indicating that knowledge can be extracted from the EBMs. This implies that not only can the distribution $p_{\phi_t}(y|x)$ learned by the teacher model be useful, but the data distribution $p(x)$ can also be learned. Figure 6 provides visualizations of the performance and convergence of our GenKD.

To evaluate the performance of DFKD, we use the same teacher and student networks as in the previous experiments. The last column of Table 4 presents the quantitative result of DFKD on the CIFAR100 dataset, and figures 6b and 6e show the student accuracy for different sampling steps K. Both results demonstrate that the student accuracy increases with increasing K in class-conditional samples. Figures 6f and 6c depict the convergence of $\mathcal{L}_c$ with different sampling strategies, and in both cases, we observe a decrease in $\mathcal{L}_c$.

VII. CONCLUSION AND FUTURE WORK

In this paper, we investigate the influence of Deep Generative Models (DGMs) on Knowledge Distillation (KD) performance. DGMs learn a data distribution with varying degrees of likelihood function smoothness. We observe that, besides the quality of the sampled pseudo samples, the smoothness of the output distribution is also crucial for KD performance. Specifically, as the log-likelihood and the quality of the sampled pseudo samples increase, the KD performance first improves and then deteriorates. We further analyze the effect of distribution smoothness through tSNE visualization of teacher features.

Besides, this paper proposes GenKD as a novel approach to improve the performance of DFKD by treating the EBM as a plug-in module. Our method updates data by minimizing the divergence between the generated data distribution p_θ and the real data distribution p_{data}, and the search process takes place in the distribution space, which makes it more robust, easier to generalize, and faster. The use of policy gradient enables us to update the parameters effectively, and the Markov property of Langevin MCMC further improves the performance. Our approach also allows for greater control over the generation process, leading to improved KD training results.

As a potential avenue for future research, we suggest two directions to improve GenKD. Firstly, it would be beneficial to investigate variants of recent DGMs such as diffusion-based EBMs [64] and VAEBMs [63]. These models could offer a tighter approximation of gradients, which may lead to further improvements in performance. Secondly, we propose that the EBM module could be applied to other downstream tasks such as zero-shot and few-shot learning. This extension would significantly expand the utility of deep generative models beyond KD.

VIII. CONFLICTS OF INTEREST

The authors declare that there is no conflict of interest regarding the publication of this paper.

REFERENCES

[1] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015, *arXiv:1503.02531*.

[2] S. Zagoruyko and N. Komodakis, "Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer," 2016, *arXiv:1612.03928*.

[3] L. Chen, D. Wang, Z. Gan, J. Liu, R. Henao, and L. Carin, "Wasserstein contrastive representation distillation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 16296–16305.

[4] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "FitNets: Hints for thin deep nets," 2014, *arXiv:1412.6550*.

[5] Q. Feng, D. He, S. Zeadally, M. K. Khan, and N. Kumar, "A survey on privacy protection in blockchain system," *J. Netw. Comput. Appl.*, vol. 126, pp. 45–58, Jan. 2019.

[6] S. Augenstein, H. B. McMahan, D. Ramage, S. Ramaswamy, P. Kairouz, M. Chen, R. Mathews, and B. A. Y. Arcas, "Generative models for effective ML on private, decentralized datasets," 2019, *arXiv:1911.06679*.

[7] G. K. Nayak, K. R. Mopuri, V. Shaj, V. B. Radhakrishnan, and A. Chakraborty, "Zero-shot knowledge distillation in deep networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 4743–4751.

[8] L. Luo, M. Sandler, Z. Lin, A. Zhmoginov, and A. Howard, "Large-scale generative data-free distillation," 2020, *arXiv:2012.05578*.

[9] J.-B. Truong, P. Maini, R. J. Walls, and N. Papernot, "Data-free model extraction," 2020, *arXiv:2011.14779*.

[10] H. Yin, P. Molchanov, J. M. Alvarez, Z. Li, A. Mallya, D. Hoiem, N. K. Jha, and J. Kautz, "Dreaming to distill: Data-free knowledge transfer via DeepInversion," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 8715–8724.

[11] K. Bhardwaj, N. Suda, and R. Marculescu, "Dream distillation: A data-independent model compression framework," 2019, *arXiv:1905.07072*.

[12] H. Chen, Y. Wang, C. Xu, Z. Yang, C. Liu, B. Shi, C. Xu, C. Xu, and Q. Tian, "Data-free learning of student networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 3514–3522.

[13] H. Chen, T. Guo, C. Xu, W. Li, C. Xu, C. Xu, and Y. Wang, "Learning student networks in the wild," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 6428–6437.

[14] G. Fang, J. Song, X. Wang, C. Shen, X. Wang, and M. Song, "Contrastive model inversion for data-free knowledge distillation," 2021, *arXiv:2105.08584*.

[15] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved training of Wasserstein GANs," 2017, *arXiv:1704.00028*.

[16] K. Lee, W. Xu, F. Fan, and Z. Tu, "Wasserstein introspective neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3702–3711.

[17] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," Tech. Rep., 2014.

[18] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 448–456.

[19] D. J. Rezende and S. Mohamed, "Variational inference with normalizing flows," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1530–1538.

[20] D. P. Kingma and P. Dhariwal, "Glow: Generative flow with invertible 1×1 convolutions," 2018, *arXiv:1807.03039*.

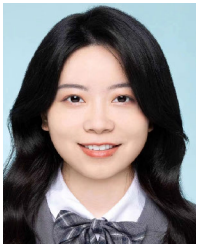
[21] Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F. Huang, "A tutorial on energy-based learning," *Predicting Structured Data*, vol. 1, Aug. 2006.

[22] Y. Du and I. Mordatch, "Implicit generation and generalization in energy-based models," 2019, *arXiv:1903.08689*.
[23] E. Nijkamp, M. Hill, S.-C. Zhu, and Y. N. Wu, "Learning non-convergent non-persistent short-run MCMC toward energy-based model," 2019, *arXiv:1904.09770*.
[24] Y. Du, S. Li, J. Tenenbaum, and I. Mordatch, "Improved contrastive divergence training of energy based models," 2020, *arXiv:2012.01316*.
[25] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," 2020, *arXiv:2006.11239*.
[26] A. Nichol and P. Dhariwal, "Improved denoising diffusion probabilistic models," 2021, *arXiv:2102.09672*.
[27] T. Karras, M. Aittala, T. Aila, and S. Laine, "Elucidating the design space of diffusion-based generative models," 2022, *arXiv:2206.00364*.
[28] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 10674–10685.
[29] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," 2020, *arXiv:2010.02502*.
[30] Y. Song, S. Garg, J. Shi, and S. Ermon, "Sliced score matching: A scalable approach to density and score estimation," in *Proc. Conf. Uncertainty Artif. Intell.*, 2019, pp. 574–584.
[31] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," 2020, *arXiv:2011.13456*.
[32] Y. Zhang, T. Xiang, T. M. Hospedales, and H. Lu, "Deep mutual learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4320–4328.
[33] T. Furlanello, Z. C. Lipton, M. Tschannen, L. Itti, and A. Anandkumar, "Born again neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1607–1616.
[34] S. I. Mirzadeh, M. Farajtabar, A. Li, N. Levine, A. Matsukawa, and H. Ghasemzadeh, "Improved knowledge distillation via teacher assistant," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 5191–5198.
[35] C. Yang, L. Xie, C. Su, and A. L. Yuille, "Snapshot distillation: Teacher-student optimization in one generation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2854–2863.
[36] J. H. Cho and B. Hariharan, "On the efficacy of knowledge distillation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Jun. 2019, pp. 4793–4801.
[37] B. Zhao, Q. Cui, R. Song, Y. Qiu, and J. Liang, "Decoupled knowledge distillation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 11943–11952.
[38] B. Heo, J. Kim, S. Yun, H. Park, N. Kwak, and J. Y. Choi, "A comprehensive overhaul of feature distillation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1921–1930.
[39] B. Heo, M. Lee, S. Yun, and J. Y. Choi, "Knowledge transfer via distillation of activation boundaries formed by hidden neurons," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 3779–3787.
[40] Z. Huang and N. Wang, "Like what you like: Knowledge distill via neuron selectivity transfer," 2017, *arXiv:1707.01219*.
[41] J. Kim, S. Park, and N. Kwak, "Paraphrasing complex network: Network compression via factor transfer," 2018, *arXiv:1802.04977*.
[42] J. Yim, D. Joo, J. Bae, and J. Kim, "A gift from knowledge distillation: Fast optimization, network minimization and transfer learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 7130–7138.
[43] F. Tung and G. Mori, "Similarity-preserving knowledge distillation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1365–1374.
[44] B. Peng, X. Jin, D. Li, S. Zhou, Y. Wu, J. Liu, Z. Zhang, and Y. Liu, "Correlation congruence for knowledge distillation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 5006–5015.
[45] K. I. Lee, S. Lee, and B. C. Song, "Zero-shot knowledge distillation using label-free adversarial perturbation with Taylor approximation," *IEEE Access*, vol. 9, pp. 45454–45461, 2021.
[46] Z. Wang, "Zero-shot knowledge distillation from a decision-based black-box model," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 10675–10685.
[47] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.
[48] K. Binici, N. T. Pham, T. Mitra, and K. Leman, "Preventing catastrophic forgetting and distribution mismatch in knowledge distillation via synthetic data," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2022, pp. 663–671.
[49] K. Binici, S. Aggarwal, N. T. Pham, K. Leman, and T. Mitra, "Robust and resource-efficient data-free knowledge distillation by generative pseudo replay," 2022, *arXiv:2201.03019*.
[50] W. Grathwohl, K.-C. Wang, J.-H. Jacobsen, D. Duvenaud, M. Norouzi, and K. Swersky, "Your classifier is secretly an energy based model and you should treat it like one," 2019, *arXiv:1912.03263*.
[51] R. Boney, J. Kannala, and A. Ilin, "Regularizing model-based planning with energy-based models," in *Proc. Conf. Robot Learn.*, 2020, pp. 182–191.
[52] Y. Deng, A. Bakhtin, M. Ott, A. Szlam, and M. Ranzato, "Residual energy-based models for text generation," 2020, *arXiv:2004.11714*.
[53] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," 2013, *arXiv:1312.6114*.
[54] K. Sohn, H. Lee, and X. Yan, "Learning structured output representation using deep conditional generative models," in *Proc. NIPS*, 2015, pp. 1–9.
[55] C. P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, and A. Lerchner, "Understanding disentangling in β-VAE," 2018, *arXiv:1804.03599*.
[56] A. Kumar, P. Sattigeri, and A. Balakrishnan, "Variational inference of disentangled latent concepts from unlabeled observations," 2017, *arXiv:1711.00848*.
[57] J. S. Liu and J. S. Liu, *Monte Carlo Strategies in Scientific Computing*, vol. 10. Berlin, Germany: Springer, 2001.
[58] J. Hammersley, *Monte Carlo Methods*. Berlin, Germany: Springer, 2013.
[59] R. E. Caflisch, "Monte Carlo and quasi-Monte Carlo methods," *Acta Numer.*, vol. 7, pp. 1–49, Jan. 1998.
[60] C. Andrieu, A. Doucet, and R. Holenstein, "Particle Markov chain Monte Carlo methods," *J. Roy. Stat. Soc. B, Stat. Methodol.*, vol. 72, no. 3, pp. 269–342, 2010.
[61] A. Vahdat, K. Kreis, and J. Kautz, "Score-based generative modeling in latent space," in *Proc. Neural Inf. Process. Syst.*, 2021, pp. 11287–11302.
[62] J. Xie, Z. Zheng, X. Fang, S.-C. Zhu, and Y. N. Wu, "Cooperative training of fast thinking initializer and slow thinking solver for conditional learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 8, pp. 3957–3973, Aug. 2022.
[63] T. Han, E. Nijkamp, L. Zhou, B. Pang, S.-C. Zhu, and Y. N. Wu, "Joint training of variational auto-encoder and latent energy-based model," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 7978–7987.
[64] R. Gao, Y. Song, B. Poole, Y. N. Wu, and D. P. Kingma, "Learning energy-based models by diffusion recovery likelihood," 2020, *arXiv:2012.08125*.
[65] A. Jahanian, X. Puig, Y. Tian, and P. Isola, "Generative models as a data source for multiview representation learning," 2021, *arXiv:2106.05258*.
[66] A. Mordvintsev, C. Olah, and M. Tyka, "Inceptionism: Going deeper into neural networks," Tech. Rep., 2015.
[67] X. Yang and S. Ji, "JEM++: Improved techniques for training JEM," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 6494–6503.
[68] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2016, *arXiv:1511.06434*.
[69] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4396–4405.
[70] I. Higgins, L. Matthey, A. Pal, C. P. Burgess, X. Glorot, M. M. Botvinick, S. Mohamed, and A. Lerchner, "beta-VAE: Learning basic visual concepts with a constrained variational framework," in *Proc. Int. Conf. Learn. Represent.*, 2017.
[71] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.
[72] T. Salimans, I. J. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training GANs," 2016, *arXiv:1606.03498*.
[73] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANs trained by a two time-scale update rule converge to a local Nash equilibrium," in *Proc. NIPS*, 2017, pp. 1–12.
[74] M. Welling and Y. W. Teh, "Bayesian learning via stochastic gradient Langevin dynamics," in *Proc. 28th Int. Conf. Mach. Learn.*, 2011, pp. 681–688.

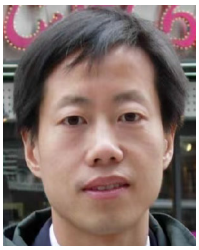
[75] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, nos. 3–4, pp. 229–256, 1992.
[76] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2000, pp. 1057–1063.
[77] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing Atari with deep reinforcement learning," 2013, *arXiv:1312.5602*.
[78] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, and A. Desmaison, "PyTorch: An imperative style, high-performance deep learning library," 2019, *arXiv:1912.01703*.
[79] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Tech. Rep., 2009.
[80] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *Proc. NIPS*, 2011, pp. 1–9.
[81] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

JINGRU LI received the B.S. degree majored in computer science from Zhejiang University, Hangzhou, Zhejiang, China, where he is currently pursuing the Ph.D. degree with the College of Computer Science and Technology. His current research interests include image inpainting, knowledge distillation, and data-free knowledge distillation.

XIAOFENG CHEN received the M.S. degree majored in computer science and technology from Sichuan University. He is currently pursuing the Ph.D. degree with the College of Computer Science and Technology, Zhejiang University, Hangzhou, Zhejiang, China. He is also the Technical Standards Director of Hangzhou Qulian Technology. His current research interests include blockchain and DLT, its application, and standardization.



PEIYU ZHENG received the M.S. degree majored in software engineering from Zhejiang University. She is currently with Hangzhou Qulian Technology Company Ltd., Hangzhou, Zhejiang, China. Her current research interests include blockchain technology and its standardization.



QIANG WANG received the B.S. and M.S. degrees in mathematics and the Ph.D. degree in computer science from Zhejiang University, China, in 1990 and 1993, respectively. He is currently an Associate Professor with the College of Computer Science, Zhejiang University. His current research interests include artificial intelligence, blockchain, and digital image processing.

ZHI YU received the Ph.D. degree majored in computer science from the College of Computer Science and Technology, Zhejiang University, Hangzhou, Zhejiang, China. He is currently with the Zhejiang Provincial Key Laboratory of Service Robot, College of Computer Science, Zhejiang University. His current research interests include data mining, accessibility, and optimization theory.