2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) | 979-8-3503-5300-6/24/$31.00 ©2024 IEEE | DOI: 10.1109/CVPR52733.2024.02252

NAYER: Noisy Layer Data Generation for Efficient and Effective Data-free Knowledge Distillation

Minh-Tuan Tran[1], Trung Le[1], Xuan-May Le[2], Mehrtash Harandi[1], Quan Hung Tran[3], Dinh Phung[1]
[1]Monash University, [2]University of Melbourne, [3]ServiceNow
{tuan.tran7,trunglm,mehrtash.harandi,dinh.phung}@monash.edu
xuanmay.le@student.unimelb.edu.au, hungquan.tran@servicenow.com

Abstract

Data-Free Knowledge Distillation (DFKD) has made significant recent strides by transferring knowledge from a teacher neural network to a student neural network without accessing the original data. Nonetheless, existing approaches encounter a significant challenge when attempting to generate samples from random noise inputs, which inherently lack meaningful information. Consequently, these models struggle to effectively map this noise to the ground-truth sample distribution, resulting in prolonging training times and low-quality outputs. In this paper, we propose a novel Noisy Layer Generation method (NAYER) which relocates the random source from the input to a noisy layer and utilizes the meaningful constant label-text embedding (LTE) as the input. LTE is generated by using the language model once, and then it is stored in memory for all subsequent training processes. The significance of LTE lies in its ability to contain substantial meaningful inter-class information, enabling the generation of high-quality samples with only a few training steps. Simultaneously, the noisy layer plays a key role in addressing the issue of diversity in sample generation by preventing the model from overemphasizing the constrained label information. By reinitializing the noisy layer in each iteration, we aim to facilitate the generation of diverse samples while still retaining the method's efficiency, thanks to the ease of learning provided by LTE. Experiments carried out on multiple datasets demonstrate that our NAYER not only outperforms the state-of-the-art methods but also achieves speeds 5 to 15 times faster than previous approaches. The code is available at https://github.com/tmtuan1307/nayer.

1. Introduction

Knowledge distillation (KD) [12, 26, 29, 36, 41] aims to train a student model capable of emulating the capabilities of a pre-trained teacher model. Over the past decade, KD has been explored across diverse domains, including image recognition [25, 30], software engineering [10], and natural language processing [33]. Conventional KD methods generally assume that the student model has access to all or part of the teacher's training data. However, real-world applications often impose constraints on accessing the original training data. This issue becomes particularly relevant in cases of privacy-sensitive medical data, which may contain personal information or data considered proprietary by vendors. Consequently, in such contexts, conventional KD methods no longer suffice to address the challenges posed.

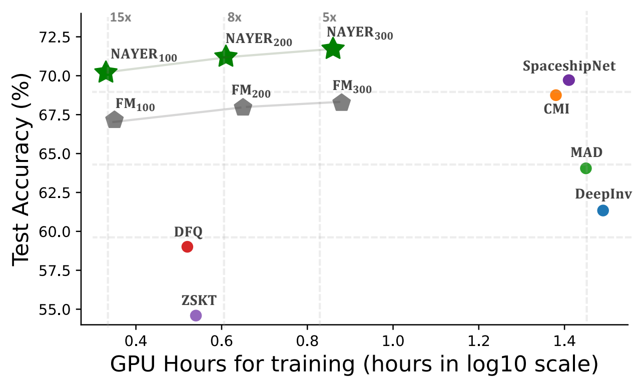

Figure 1. Accuracy of student models and GPU hours of training time on CIFAR-100 dataset. All variants of our method NAYER not only attains the highest accuracies across but also accelerates the training process by 5 to 15 times compared to DeepInv [37].

Data-Free Knowledge Distillation (DFKD) [1, 6, 8, 28, 35, 37, 39] has recently seen significant advancements as an alternative method. Its core principle involves transferring knowledge from a teacher neural network ($\mathcal{T}$) to a student neural network ($\mathcal{S}$) by generating synthetic data instead of accessing the original training data. The synthetic data enable adversarial training of the generator and student [21, 24]. In this setup, the student seeks to match the teacher's predictions on synthetic data, while the generator aims to create samples that maximize the discrepancy be-

2575-7075/24/$31.00 ©2024 IEEE
DOI 10.1109/CVPR52733.2024.02252

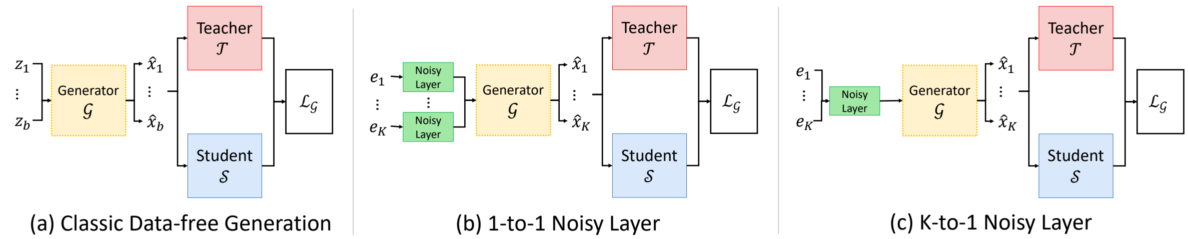

Figure 2. Data Generation Strategies: (a) Classic method which optimizes random noise (z); (b) Using one noisy layer for generating one synthetic image from the label-text embedding ($\boldsymbol{e_y}$); (c) Using one noisy layer to generate multiple synthetic images.

tween the student's and teacher's predictions (Figure 2a).

Due to its reliance on synthetic samples, the need for an effective and efficient data-free generation technique becomes imperative. A major limitation of current DKFD methods is that they merely generate synthetic samples from random noise, neglecting to incorporate supportive and semantic information [2, 7, 28, 39]. This limitation in turn incurs the generation of low-quality data or excessive time requirements for training the generator, rendering them unsuitable for large-scale tasks. Notably, almost state-of-the-art (SOTA) DFKD methods do not report results on large-scale ImageNet due to the significant training time involved. Even with smaller datasets such as CIFAR-100 (see Figure 1), SOTA DFKD methods such as CMI [8], MAD [6], or DeepInv still demand approximately 25 to 30 hours of training while struggling to achieve high accuracy. This emphasizes the pressing need for more efficient and effective DFKD techniques.

To address mentioned problem, we introduce a simple yet effective DFKD method called Noisy lAYER generation (NAYER). Our approach relocates the source of randomness from the input to the noisy layer and utilizes the meaningful label-text embedding (LTE) generated by a pretrained language model (LM) [14, 31, 32] as the input. In this context, LTE plays a crucial role in accelerating the training process due to its ability to encapsulate useful interclass information. Note that, there is a common observation that the text with similar meanings tend to exhibit closer embedding proximity to one another [16, 17, 22]. For instance, the text embedding of sentence *"A class of a dog"* and *"A class of a cat"* is always closer compared to *"A class of a car"*. Consequently, by using LTE as input, our approach can proficiently generate high-quality samples that closely mimic the distributions of their respective classes with only a few training steps. It is important to note that our method only queries the LTE from the LM once. This LTE is then stored in memory for subsequent processing, and we do not use the language model in the training process.

However, when utilizing a constant LTE as the input, we empirically observed that the generator consistently produces a set of similar data lacking diversity in every iteration. Our solution addresses this issue by introducing the layer-level random source by adding a noisy layer (NL) to learn the constant label information. This involves incorporating a random NL to function as an intermediary between the generator and LTE, which prevents the generator from relying solely on unchanging label information (Figure 2b). The source of randomness now comes from the random reinitialization of the NL for each iteration. Through this mechanism, we aim to effectively mitigate the risk of overemphasizing label information, thus enhancing the diversity of synthesized images. Furthermore, thanks to the inherent ease of learning label-text embeddings, regardless of how it is initialized, jointly training the NL with generator can consistently generate high-quality samples in just a few steps, thereby maintaining the method's efficiency. Additionally, we propose leveraging a single NL to generate multiple samples (e.g., 100 images across 100 classes of CIFAR-100) (Figure 2c). This strategy reduces the number of training parameters and enhances diversity by leveraging multiple gradient sources from various classes.

Our major contributions are summarized as follows:

- We propose NAYER, a simple yet effective DFKD method based on LTE and a noisy layer, providing the fast training with high classification performance.
- We introduce a K-to-1 noisy layer, which utilizes only a single noisy layer to generate multiple samples.
- Experiments on CIFAR100, TinyImageNet and ImageNet, demonstrate that our method ourperforms SOTA algorithms in both accuracy and training time. Specifically, our methods achieve speeds that are 5 to even 15 times faster while also attaining higher accuracies compared to previous methods (Figure 1).
- To the best of our knowledge, we are the first to introduce the use of LTE and the concept of layer-level random source for DFKD.

2. Related Work

Data-Free Knowledge Distillation. DFKD methods [6, 8, 28, 37, 39] generate synthetic images to transfer knowledge from a pre-trained teacher model to a student model. These data are used to jointly train the generator and the student in an adversarial manner [21]. In this adversarial learning scheme, the student aims to make predictions close to the teacher's on synthetic data, while the generator endeavors to create samples that align with the teacher's confidence but also maximize the mismatch between the student's and the teacher's predictions. This adversarial game enables an rapid exploration of synthetic distributions useful for knowledge transfer between the teacher and the student.

Data-Free Generation. As the central principle of DFKD revolves around synthetic samples, the data-free generation technique plays a pivotal role. [37] proposes the image-optimized method which attempts to optimize the random noise images using teacher network batch normalization statistics. Sample-optimized methods [8, 39] focus on optimizing random noise over numerous training steps to produce synthetic images in case-by-case strategy. In contrast, generator-optimized methods [2, 6, 28] attempt to ensure that the generator has the capacity to comprehensively encompass the entire distribution of the original data. In the other words, regardless of the input random noise, these methods aim to consistently yield high-quality samples for training the student model. This approach often prolongs the training process and may not consistently produce high-quality samples, particularly when diverse noises are employed during both the sampling and training phases. Furthermore, the main problem in existing data-free generation is the use of random noise input without any meaningful information, leading to generate the low-quality samples and prolonged training times for the generator. [7] introduced FM, a method incorporating a meta generator to accelerate the DFKD process significantly. However, this acceleration comes at the cost of a noticeable trade-off in accuracy.

Synthetizing Samples from Label Information. Drawing inspiration from the success of incorporating label information in adversarial frameworks like Conditional GAN [19, 23, 27], several DFKD methods have adopted strategies to generate images guided by labels. In these approaches, a common practice involves fusing random noise ($\boldsymbol{z}$) with a learnable embedding ($\boldsymbol{e_y}$) of the one-hot label vector, which is used as input for the model [6, 20, 38]. This combination enhances control over the resulting class-specific synthetic images. However, despite the potential of label information, its application has yielded only minor improvements. This can be attributed to two key factors. Firstly, the one-hot vector introduces sparse information that merely distinguishes labels uniformly, failing to capture the nuanced relationships between different classes. Consequently, the model struggles to generate images that align closely with ground-truth distributions. Secondly, there exists a challenge in balancing the generated images' quality and diversity when incorporating label information. This can inadvertently lead to an overemphasis on label-related details, potentially overshadowing the crucial contribution of random noise, which is necessary for generating a diverse range of samples

3. Proposed Method

3.1. Problem Formulation

Consider a training dataset $D = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^m$ with $\boldsymbol{x}_i \in \mathbb{R}^{c\times h\times w}$ and $\boldsymbol{y}_i \in \{1, 2, \cdots, K\}$, where the pair $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ represents a training sample and its corresponding label, respectively. Let $\mathcal{T} = \mathcal{T}_{\theta_\mathcal{T}}$ be a pre-trained teacher network on D. The objective of DFKD is to train a student network $\mathcal{S} = \mathcal{S}_{\theta_\mathcal{S}}$ to emulate $\mathcal{T}$'s performance, all without needing access to the original dataset D.

To achieve this, we employ the lightweight generator $\mathcal{G}_{\theta_\mathcal{G}}$ to generate synthetic images and subsequently use them to train a student network $\mathcal{S}$. Specifically, in contrast to existing DFKD methods [6–8, 28, 37, 39], our approach utilizes a meaningful constant label-text embedding (LTE) as the input for $\mathcal{G}$ instead of random noise. Due to LTE's capability to encapsulate valuable interclass information, this accelerates the generation process, expediting the training time (Section 3.2). Following that, we propose the use of a layer-level random source (Noisy Layer) to better adapt with LTE for generating diverse synthetics (Section 3.3). Finally, the synthetic images are employed for the joint training of the generator and student in an adversarial manner to enhance knowledge transfer (Section 3.4).

3.2. Label-Text Embedding as Generator's Input

The main limitation of existing DFKD methods is synthetize data from random noise, which have no supportive and semantic information. Therefore, they usually generate very low-quality data [7] or require a excessive training time for high quality image generation [6, 8, 28, 37, 39]. There are also several methods use to one-hot vector of classes as the additional input to resemble the conditional generator, however its application has yielded only minor improvements. The main reason is the one-hot vector (OH) introduces sparse information which make a generator hard to learn about it. Furthermore, OH merely distinguishes labels uniformly, failing to capture the nuanced relationships between different classes.

To address this problem, we are the first to propose the use of label-text embeddings for DFKD by employing them as an input for the generator. LTE, as a dense vector with richer information, facilitates an easier learning process for the model. Additionally, LTE capitalizes on the tendency for text with similar meanings to exhibit proximity in their embeddings [14]. Figure 3a-c visually represents the LTE, highlighting their superior capacity to depict the relationship between the 'Dog' and 'Cat' classes. This is evident in their closer proximity (shorter distance) when compared to the 'Car' class. This characteristic of LTE contributes

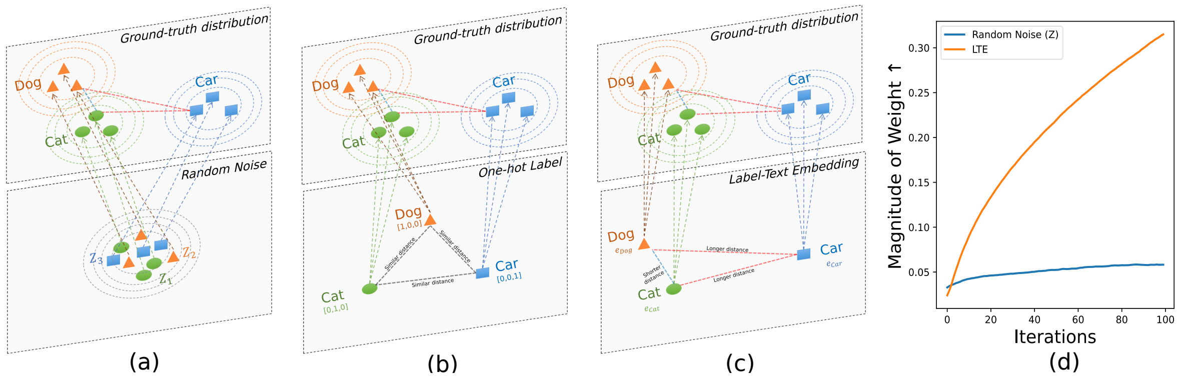

Figure 3. (a) Random noise for data generation. (b) One-hot labels only uniformly distinguish labels, lacking inter-class relationships. In contrast, (c) LTE captures inter-class connections, bringing similar classes closer in the embedding space. This proximity enhances the similarity between the input and ground-truth sample distributions, thereby allowing the model to more easily mimic the ground-truth distribution and accelerating the learning process. (d) The averaging magnitude of weight used to learn LTE is much larger than those for random noise, highlighting the model's negative focus on label information while ignoring random noise.

to making the input distribution (representing labels) and ground-truth distribution (representing actual data) more similar. As a result, it facilitates the model's mapping between these two distributions, accelerating the learning process and generating high-quality images.

Prompt Engineering. Given the list of all classes $\boldsymbol{y} = [\boldsymbol{y}_1, \cdots, \boldsymbol{y}_K]$, their label text $Y_{\boldsymbol{y}} = [Y_{\boldsymbol{y}_1}, \cdots, Y_{\boldsymbol{y}_K}]$ is generated by using a manually designed prompt template such as "`a class of a {class_name}`". Then, the label-text prompt is then embedded using a pre-trained text encoder $\mathcal{C}$ as follows:

$$\boldsymbol{e}_{\boldsymbol{y}} = \mathcal{C}(Y_{\boldsymbol{y}}) \ . \tag{1}$$

LTE Pool. Importantly, the embedding $\boldsymbol{e}_{\boldsymbol{y}}$ is generated once and then stored in the LTE pool $\mathcal{P}$, remaining fixed throughout the entire training process. The text encoder $\mathcal{C}$ is not utilized during the training process. In training phases, with a batch of pseudo-labels $\hat{\boldsymbol{y}}$, we retrieve their corresponding LTEs from $\boldsymbol{e}_{\hat{\boldsymbol{y}}} \sim \mathcal{P}$ and employ these LTEs as inputs for the generator. This eliminates the reliance on random noise for synthetic image generation.

$$\hat{\boldsymbol{x}} = \mathcal{G}(\boldsymbol{e}_{\hat{\boldsymbol{y}}}) \ . \tag{2}$$

We conducted an ablation study to analyze the impact of different prompt engineering template and language model (LM) for generating LTEs in Section 4.4.

Thanks to the informative content embedded in LTE, our approach can efficiently produce high-quality samples with minimal computational steps. We have also conducted an empirical study to substantiate this claim, as illustrated in Table 4. The results of this study highlight that LTE significantly accelerates convergence in terms of Cross-Entropy (CE) Loss and yields higher-quality images (as measured by the Inception Score or IS score) compared to random noise and one-hot vectors. This acceleration empowers our method to achieve convergence with a considerably smaller training steps for generator (30 steps for CIFAR10 and 40 steps for CIFAR100), compared to the 2,000 steps required by DeepInv or the 500 steps of CMI, all while maintaining superior accuracy (as detailed in Table 1).

3.3. Generating Diverse Samples with Noisy Layer

While leveraging label information provides advantages for data generation, the synthetic images are less diverse due to the absence of a random source. Two common solutions involve concatenating random noise $\boldsymbol{z}$ and $\boldsymbol{e}_{\boldsymbol{y}}$ or using their sum as the generator input. However, both approaches have limitations. Concatenation raises the risk of overemphasizing label-related information, as evidenced by significantly larger weight magnitudes for learning LTE compared to random noise (Figure 3d), which can be seen as the significance of these weights [9]. Using the sum of $\boldsymbol{v} = \boldsymbol{e}_{\hat{\boldsymbol{y}}} + \beta \boldsymbol{z}$ faces challenges: a low β results in an insufficient random source for diverse sampling, and a high β may overshadow LTE features, leading to a reliance on random noise $\boldsymbol{z}$. This challenge is also observed in some existing methods [6, 20], where the application of the sum of noise and label information provides minimal improvement compared to an unconditional generator.

To effectively introduce randomness to LTE, we propose the concept of a layer-level random source with the Noisy Layer. The source of randomness now stems from the random reinitialization of the NL during each iteration. With each different initialization, the NL learns LTE in a distinct way, successfully mitigating the risk of a negative bias towards LTE. Unlike existing sources of randomness, the design of NL provides a larger random parameter to enhance the diversity of the synthesized images. Furthermore, due to the straightforward training of LTE, regardless of its initialization, the joint training of the noisy layer and the generator consistently yields high-quality samples within a few iterations, thus preserving the method's efficiency.

Noisy Layer Architecture. We design the NL $\mathcal{Z}_{\theta_{\mathcal{Z}}}$ as a combination of a `BatchNorm` layer and a single `Linear` layer. The input size of the `Linear` layer matches the embedding size of the text encoder (e), and the output size corresponds to the noise dimension (r). Typically, this output size is set to 1,000, following to [6,28,39]. The simplicity of the single `Linear` layer is crucial for expediting the generation process. It converges rapidly without requiring an excessive number of steps, yet its size remains sufficiently large to provide an sufficient random source for the generator. Additionally, a `BatchNorm` module plays a role in increasing the distance between LTEs (from averaging 0.015 to 0.45 using L2 distance), helping the model discriminate these LTEs easier and thereby speeding up the training process. Furthermore, with a different batch of $\hat{\boldsymbol{y}}$, the output of `BatchNorm` can vary, introducing a slight additional randomness for the generator. The ablation study analyzing the impact of different architectures of NL can be found in the **Supplemental Material**.

Given LTEs $\boldsymbol{e}_{\hat{\boldsymbol{y}}}$, we feed these $\boldsymbol{e}_{\hat{\boldsymbol{y}}}$ into the noisy layer $\mathcal{Z}$. Then, the output of the noisy layer is fed into the generator $\mathcal{G}$ to produce the batch of synthetic images $\hat{\boldsymbol{x}}$:

$$\mathcal{Z}(\boldsymbol{e}_{\hat{\boldsymbol{y}}}) = \texttt{Linear}(\texttt{BatchNorm}(\boldsymbol{e}_{\hat{\boldsymbol{y}}}))\ . \tag{3}$$

$$\hat{\boldsymbol{x}} = \mathcal{G}(\mathcal{Z}(\boldsymbol{e}_{\hat{\boldsymbol{y}}}))\ . \tag{4}$$

K-to-1 Noisy Layer. In the existing approach, a separate random source is created for each instance, as similar inputs generate similar samples. In contrast, we propose employing a single noisy layer to learn from all available classes (K-to-1) by inputting $\boldsymbol{e}_{\hat{\boldsymbol{y}}}$ with $\hat{\boldsymbol{y}} = 1, \ldots, K$ to a single noisy layer $\mathcal{Z}$. This design enables the noisy layer to generate multiple samples simultaneously, such as a maximum of 100 for CIFAR100 or 10 for CIFAR10, thus reducing a parameter size and efficiently expediting training. The underlying idea revolves around the fact that each class has distinct LTEs. Thus, by supplying different inputs of $\boldsymbol{e}_{\hat{\boldsymbol{y}}}$ from K classes, the noisy layer can still generate diverse images. Furthermore, we also empirically observe that using a single noisy layer to synthesize a batch of images (K-to-1) enriches generator diversity, ensuring both fast convergence and high-quality sample generation. This enhancement can be attributed to the use of multiple gradient sources from diverse classes, which can further enriches the diversity of the noisy layer's output.

3.4. Generator and Student Updating

To make it easier to follow, we provide the architecture of NAYER in Figure 4 and the detailed pseudocode in Algorithm 1, wherein NAYER initially embeds all label text using a text encoder. Subsequently, our method undergoes

Algorithm 1: NAYER

Input: pre-trained teacher $\mathcal{T}_{\theta_{\mathcal{T}}}$, student $\mathcal{S}_{\theta_{\mathcal{S}}}$, generator $\mathcal{G}_{\theta_{\mathcal{G}}}$, text encoder $\mathcal{C}_{\theta_{\mathcal{C}}}$, list of labels $\boldsymbol{y}$ and list of text of these labels $Y_{\boldsymbol{y}}$;

Output: An optimized student $\mathcal{S}_{\theta_{\mathcal{S}}}$

1 Initializing $\mathcal{P} = \{\}, \mathcal{M} = \{\}$;
2 Store all embeddings $\boldsymbol{e}_{\boldsymbol{y}} = \mathcal{C}(Y_{\boldsymbol{y}})$ into $\mathcal{P}$;
3 **for** $\mathcal{E}$ *epochs* **do**
4   **for** I *iterations* **do**
5   Randomly reinitializing noisy layers $\mathcal{Z}_{\theta_{\mathcal{Z}}}$ and pseudo label $\hat{\boldsymbol{y}}$ for each iteration;
6   Query $\boldsymbol{e}_{\hat{\boldsymbol{y}}} \sim \mathcal{P}$;
7   **for** g *steps* **do**
8             $\hat{\boldsymbol{x}} \leftarrow \mathcal{G}(\mathcal{Z}(\boldsymbol{e}_{\hat{\boldsymbol{y}}}))$;
9             $\mathcal{L}_{\mathcal{Z}} \leftarrow \alpha_{cls}\mathcal{L}_{\text{CE}}(\mathcal{T}(\hat{\boldsymbol{x}}), \hat{\boldsymbol{y}}) - \alpha_{adv}\mathcal{L}_{\text{KL}}(\mathcal{T}(\hat{\boldsymbol{x}}), \mathcal{S}(\hat{\boldsymbol{x}})) + \alpha_{bn}\mathcal{L}_{\text{BN}}(\mathcal{T}(\hat{\boldsymbol{x}}))$;
10   Update $\theta_{\mathcal{G}}, \theta_{\mathcal{Z}}$ by minimizing $\mathcal{L}_{\mathcal{Z}}$;
11         $\mathcal{M} \leftarrow \mathcal{M} \cup \hat{\boldsymbol{x}}$;
12   **for** S *iterations* **do**
13         $\hat{\boldsymbol{x}} \sim \mathcal{M}$;
14   Update $\theta_{\mathcal{S}}$ by minimizing $\mathcal{L}_{\mathcal{S}} \leftarrow \mathcal{L}_{\text{KL}}(\mathcal{T}(\hat{\boldsymbol{x}}), \mathcal{S}(\hat{\boldsymbol{x}}))$;

training for $\mathcal{E}$ epochs. Within each training epoch, NAYER consists of two distinct phases. The first phase involves training the generator. In each iteration I, as described in Algorithm 1, the noisy layer $\mathcal{Z}$ is reinitialized (line 5) before being utilized to learn the LTE. The generator and the noisy layer are then trained through g steps using Eq. (5) to optimize their performance (line 10).

$$\min_{\theta_{\mathcal{G}}, \theta_{\mathcal{Z}}} \mathcal{L}_{\mathcal{Z}} \triangleq \mathbb{E}_{\hat{\boldsymbol{x}} \sim \mathcal{G}(\mathcal{Z}(\boldsymbol{e}_{\hat{\boldsymbol{y}}}))} \Big[\alpha_{cls}\mathcal{L}_{\text{CE}}(\mathcal{T}(\hat{\boldsymbol{x}}), \hat{\boldsymbol{y}}) - \alpha_{adv}\mathcal{L}_{\text{KL}}(\mathcal{T}(\hat{\boldsymbol{x}}), \mathcal{S}(\hat{\boldsymbol{x}})) + \alpha_{bn}\mathcal{L}_{\text{BN}}(\mathcal{T}(\hat{\boldsymbol{x}})) \Big]\ . \tag{5}$$

Within this context, $\mathcal{L}_{\text{CE}}$ represents the Cross-Entropy loss term, serving the purpose of training the student on images residing within the high-confidence region of the teacher's knowledge. Conversely, the negative $\mathcal{L}_{\text{KL}}$ term facilitates the exploration of synthetic distributions, boosting effective knowledge transfer between the teacher and the student. In other words, the student network takes on a role as a discriminator in GANs, ensuring the generator is geared towards producing images that the teacher has mastered, yet the student network has not previously learned. This approach facilitates the focused development of the student's understanding in areas where it lags behind the teacher, enhancing the overall knowledge transfer process. We also use batch norm regularization ($\mathcal{L}_{\text{BN}}$) [7, 37], a commonly used loss in DFKD, to constrain the mean and variance of the feature at the `BatchNorm` layer to be consistent with the running-mean and running-variance of the same layer.

The second phase involves training the student networks. During this phase, all the generated samples are stored in

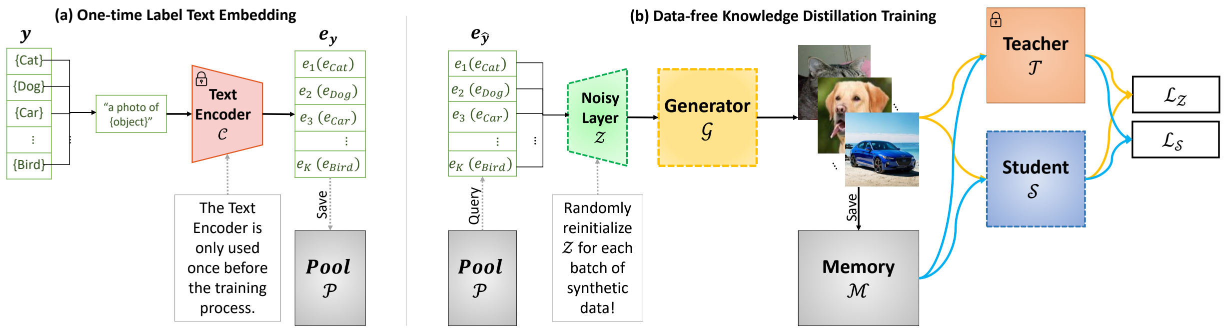

Figure 4. General Architecture of Noisy Layer Generation for Data-free Knowledge Distillation: NAYER initially employs the text encoder to generate the LTEs, which are then stored in the memory pool for model training. In each training batch, the LTEs serve as input for the noisy layer $\mathcal{Z}$ and generator $\mathcal{G}$ to produce synthetic images. Finally, these images are used for the joint training of the generator, noisy layer, and student network using Eq. 5 and Eq. 6.

the memory module $\mathcal{M}$ to mitigate the risk of forgetting (line 10), following a similar approach as outlined in [7]. Ultimately, the student model is trained by Eq. (6) over S iterations, utilizing the samples from $\mathcal{M}$ (lines 13 and 14).

$$\min_{\theta_{\mathcal{S}}} \mathcal{L}_{\mathcal{S}} \triangleq \mathbb{E}_{\hat{x}\sim\mathcal{M}}\left[\mathcal{L}_{\mathrm{KL}}(\mathcal{T}_{\theta_{\mathcal{T}}}(\hat{x}), \mathcal{S}_{\theta_{\mathcal{S}}}(\hat{x}))\right]. \quad (6)$$

4. Experiments

4.1. Experimental Settings

We conducted a comprehensive evaluation of our method across various backbone networks, namely ResNet [11], VGG [34], and WideResNet (WRN) [40], spanning three distinct classification datasets: CIFAR10, CIFAR100 [13], and Tiny-ImageNet [15]. The datasets feature varying scales and complexities, offering a well-rounded assessment of our method's capabilities. In detail, CIFAR10 and CIFAR100 encompass a total of 60,000 images, partitioned into 50,000 for training and 10,000 for testing. CIFAR10 comprises 10 categories, while CIFAR100 boasts 100 categories. The images within both datasets are characterized by a resolution of 32×32 pixels. On the other hand, Tiny-ImageNet comprises 100,000 training images and 10,000 validation images, with a higher resolution of 64 × 64 pixels. This dataset encompasses a diverse array of 200 image categories, contributing to the breadth and comprehensiveness of our evaluation.

4.2. Results and Analysis

Comparison with SOTA DFKD Methods. Table 1 displays the results of DFKD achieved by our methods and several state-of-the-art (SOTA) approaches. In general, previous methods exhibit limitations when generating images from random noise, impacting both training time and image diversity. By using LTE as the input and relocating the source of randomness from the input to the layer level, our approach provides highly diverse training images and faster running time. Notably, with 300 epochs, our method achieves SOTA performance in all comparison cases, except for the Resnet32/Resnet18 case in CIFAR10. However, it is essential to note that our method was designed in a straightforward manner, without incorporating innovative techniques found in current SOTA approaches, such as activation region constraints and feature exchange in SpaceshipNet [39], knowledge acquisition and retention meta-learning in KAKR [28], and momentum distillation in MAD [6].

Additional Experiments at Higher Resolution. To assess the effectiveness of NAYER, we conducted further evaluations on the more challenging ImageNet dataset. ImageNet comprises 1.3 million training images with resolutions of 224×224 pixels, spanning 1,000 categories. ImageNet's complexity surpasses that of CIFAR, making it a significantly more time-consuming task for data-free training. As displayed in Table 1, almost all DFKD methods refrain from reporting results on ImageNet due to their prolonged training times. Therefore, our comparison is primarily against DeepInv [37], and for the sake of a fair comparison, we re-conducted the experiments of FM [7] to align with our settings. The results clearly demonstrate that NAYER outperforms other methods in terms of accuracy, underscoring its efficacy on a large-scale dataset.

Training Time Comparison. As shown in Table 2, the NAYER model trained for 100 epochs (i.e., NAYER($\mathcal{E}$ = 100)) achieves an average speedup of 15× compared to DeepInv, while also delivering higher accuracies. This substantial speedup is attributed to the significantly fewer steps required for generating samples (30 for CIFAR-10 and 40 for CIFAR-100) compared to DeepInv's 2000 steps. As a result, DeepInv takes over 30 hours to complete training on CIFAR-10/CIFAR-100, whereas our method only requires approximately 2 hours. These results demonstrate that our method not only achieves high accuracy but also signifi-

Table 1. The distillation results of compared methods in CIFAR10 and CIFAR100. The best-performing method is highlighted in bold, and the runner-up is underlined. Additionally, we use superscripts to indicate the sources of these results: ^{a} for [7], ^{b} for [28], ^{c} for [6], ^{d} for [39], and ^{e} for our experiments. In this table, 'R' represents Resnet, 'W' corresponds to WideResnet, and 'V' stands for VGG.

	CIFAR10					CIFAR100					TinyImageNet	ImageNet
Method	**R34 R18**	**W402 W162**	**W402 W161**	**W402 W401**	**V11 R18**	**R34 R18**	**W402 W162**	**W402 W161**	**W402 W401**	**V11 R18**	**R34 R18**	**R50 R50**
Teacher	95.70	94.87	94.87	94.87	92.25	77.94	77.83	75.83	75.83	71.32	66.44	75.45
Student	95.20	93.95	91.12	93.94	95.20	77.10	73.56	65.31	72.19	77.10	64.87	75.45
DeepInv^a [37]	93.26	89.72	83.04	86.85	90.36	61.32	61.34	53.77	68.58	54.13	-	68.00
DFQ^a [5]	94.61	92.01	86.14	91.69	90.84	77.01	64.79	51.27	54.43	66.21	-	-
ZSKT^a [21]	93.32	89.66	83.74	86.07	89.46	67.74	54.59	36.60	53.60	54.31	-	-
CMI^a [8]	94.84	92.52	90.01	92.78	91.13	77.04	68.75	57.91	68.88	70.56	64.01	-
PREKD^b [2]	93.41	-	-	-	-	76.93	-	-	-	-	49.94	-
MBDFKD^b [3]	93.03	-	-	-	-	76.14	-	-	-	-	47.96	-
FM^a [7]	94.05	92.45	89.29	92.51	90.53	74.34	65.12	54.02	63.91	67.44	-	57.37^e
MAD^c [6]	94.90	92.64	-	-	-	77.31	64.05	-	-	-	62.32	-
KAKR_MB^b [28]	93.73	-	-	-	-	77.11	-	-	-	-	47.96	-
KAKR_GR^b [28]	94.02	-	-	-	-	77.21	-	-	-	-	49.88	-
SpaceshipNet^d [39]	**95.39**	93.25	90.38	93.56	92.27	77.41	69.95	58.06	68.78	71.41	64.04	-
NAYER ($\mathcal{E} = 100$)	94.03	93.48	91.12	93.57	91.34	76.29	70.20	59.26	69.89	71.10	61.71	-
NAYER ($\mathcal{E} = 200$)	94.89	93.84	91.60	94.03	91.93	77.07	71.22	61.90	70.68	71.53	63.12	-
NAYER ($\mathcal{E} = 300$)	95.21	**94.07**	**91.94**	**94.15**	**92.37**	**77.54**	**71.72**	**62.23**	**71.80**	**71.75**	**64.17**	**68.92**

Table 2. Comparing training times in hours using a single NVIDIA A100 for DFKD methods on CIFAR-10 and CIFAR-100 with the teacher/student models WRN40-2/WRN16-2. FM ($\mathcal{E} = 100, 200$, and 300) corresponds to the settings of three variants of our methods. We were unable to replicate the training times of KAKR and SpaceshipNet as they did not provide access to their source code.

	DeepInv	CMI	DFQ	ZSKT	MAD	SpaceshipNet	FM $\mathcal{E} = 100$	FM $\mathcal{E} = 200$	FM $\mathcal{E} = 300$	**NAYER** $\mathcal{E} = 100$	**NAYER** $\mathcal{E} = 200$	**NAYER** $\mathcal{E} = 300$
CIFAR10	89.72 (31.23h)	92.52 (24.01h)	92.01 (3.31h)	89.66 (3.44h)	92.64 (13.13h)	93.25 (22.35h)	91.63 (2.18h)	92.05 (3.98h)	92.31 (7.02h)	**93.48 (2.05h)**	93.84 (3.85h)	94.07 (6.78h)
CIFAR100	61.34 (31.23h)	68.75 (24.01h)	64.79 (3.31h)	54.59 (3.44h)	64.05 (26.45h)	69.95 (22.37h)	67.15 (2.23h)	67.75 (4.42h)	68.25 (7.56h)	**70.20 (2.15h)**	71.22 (4.03h)	71.72 (7.22h)
Avergaing Speed Up	1×	1.3×	9.73×	9.08×	1.78×	1.39×	14.17×	7.46×	4.29×	**14.88**×	7.93×	4.47×

cantly accelerates the model training process.

Additional Experiments in Data-free Quantization. To demonstrate the use of our data-free generation in other data-free tasks, we further conduct experiments in Data-free Quantization. We conducted a comparative analysis against ZeroQ [4], DFQ [5], and ZAQ [18]. ZeroQ retrains a quantized model using reconstructed data instead of original data, DFQ is a post-training quantization approach that utilizes a weight equalization scheme to eliminate outliers in both weights and activations, and ZAQ is the pioneering method that employs adversarial learning for data-free quantization. In this comparison, our method consistently demonstrated superior accuracy across all four scenarios.

Table 3. The results of compared methods in Data-free Quantization.

Dataset	Model	Bit	Float32	ZeroQ	DFQ	ZAQ	NAYER ($\mathcal{E} = 300$)
CIFAR10	**MobileNetV2**	W6A6	92.39	89.9	85.43	92.15	**92.23**
	VGG19	W4A8	93.49	92.69	92.66	93.06	**93.15**
CIFAR100	**Resnet20**	W5A5	69.58	65.7	59.42	67.94	**68.23**
	Resnet18	W4A4	77.38	70.25	40.35	72.67	**73.32**

4.3. Ablation Study

Effectiveness of Label-Text Embedding. We illustrate the impact of using LTE in comparison with random noise (Z) and one-hot vector (OH) as the inputs for the generator. As depicted in first three column in Table 4, LTE demonstrates significantly accelerated averaging convergence in terms of CE Loss. This phenomenon can be attributed to the principle that mapping between two distributions is simplified when they share greater similarity. However, the diversity metric for inputting label information (both LTE and OH) is notably lower than that of random noise. This outcome underscores the adverse effects of the generator overly focusing on constant label information.

Effectiveness of Noisy Layer. We analyze the impact of multiple random source strategies, including our NL(1-to-1), NL(K-to-1), NL without reinitiation (WoRI), the concatenation of LTE and random noise Z (cat), and the sum of them (sum(β)): $\boldsymbol{v} = \boldsymbol{e_y} + \beta Z$. Table 4 demonstrates that: 1) the sum of LTE and noise have a lower convergence time but higher accuracy and diversity if β is high,

Table 4. Comparison with different types of input and random sources involves accuracy, diversity metric and averaging convergence time, which is the average number of epochs the generator needs to synthesize data with Cross-Entropy (CE) Loss < 0.1. Each method undergoes 30 generation steps and runs for 100 epochs. "-" denotes that a model cannot provide any data with CE Loss < 0.1.

	OH	Z	LTE	cat	sum(0.1)	sum(0.5)	sum(1)	NL(woRI)	NL(1-to-1)	**NL(K-to-1)**	NL(woBN)
Averaging Convergence Time(↓)	28.23	-	**8.52**	10.47	10.17	25.12	-	8.68	9.67	9.53	16.58
Diversity Score(↑)	0.013	0.137	0.016	0.0132	0.021	0.036	0.127	0.016	0.138	**0.139**	0.131
Accuracy(↑)	12.35	90.14	13.52	13.29	18.92	85.72	90.15	14.82	93.42	**93.48**	91.15

making them similar to only using random noise Z. In contrast, if β is low, the convergence time is faster but accuracy and diversity are lower, similar to only using LTE. 2) Using NL boosts the generator's diversity while maintaining rapid convergence and high-quality sampling. 3) Using NL(K-to-1) results in faster convergence and a higher diversity score when compared to using one noisy layer for each individual image (1-to-1). 4) Without reinitiation, the NAYER provides almost similar results to only using LTE, thereby highlighting the effectiveness of reinitiation strategies. The further comparison with different architecture of NL can be found at Supplemental Material.

Effectiveness of `BatchNorm`. We analyze the impact of `BatchNorm` in Eq. 1. Table 4 shows that without using `BatchNorm` (WoBN), NAYER struggles in learning LTE and has lower accuracy (91.15% compared to 93.48%), attributed to the close proximity of LTEs, highlighting the effectiveness of using `BatchNorm` in our method.

4.4. Further Analysis

Comparison with Different Memory Size. In this comparison, we evaluate the accuracies of our NAYER and MBD-FKD models while varying the memory size. Note that, to ensure a fair comparison, we maintain identical generator architectures, including the additional linear layer (noisy layer for NAYER) for MBDFKD. The results demonstrate that: 1) With a bigger memory size, our method can have better performance. 2) Even with only 5k memory size, our method still outperforms the current SOTA DFKD method (90.41% compared to 90.38% of SpaceshipNet).

Table 5. The accuracies of NAYER and MBDFKD with varying the memory buffer size.

Memory buffer size	SOTA	5k	10k	20k	40k	100k	200k	Full
MBDFKD	90.38	73.33	74.12	73.72	72.68	71.96	71.27	70.72
NAYER	90.38	**90.41**	**90.76**	**90.98**	**91.21**	**91.64**	**91.86**	**91.94**

Comparison with Different Prompting Engineering Templates. We analyze the impact of different prompting engineering techniques to generate the label text. We propose three different ways to prompt the label text, including P1: `"a class of a {class_name}"`, P2: `"a photo of a {class_name}"`, P3: `"a photo of a {class_index}"`. Table 6 demonstrates that: 1) P1 and P2 has the best accuracy and far higher than current SOTA; 2) By using only the label index instead of the label name, the performance of P3 remains far better than the best baseline (93.90% and 71.37% compared to 93.25% and 69.95% for SpaceshipNet). From this, we can infer that using the label index is possible in the datasets with less meaningful labels, further showing the effectiveness of our methods in real-world applications.

Table 6. Accuracies of different prompt engineering methods.

	CIFAR-10				CIFAR-100			
Text Encoder	SOTA	P1	P2	P3	SOTA	P1	P2	P3
Accuracy	93.25	93.96	**94.07**	93.28	69.95	71.68	**71.72**	71.12

Comparison with Different Text Encoder. We evaluate our NAYER using three common text encoders—Doc2Vec [14], SBERT [32], and CLIP [31] (Table 7). Table 7 shows that: 1) Our method performs well across almost all language models, thanks to their ability to capture the relations of label-text. Furthermore, by combining any language model with the label-index prompt engineering discussed in the previous section, our method is capable of working in various domains, even without meaningful label-text. Secondly, the ability of foundational models like CLIP is demonstrated to improve our model's performance, attributed to the multimodal nature of its vision-language model. Nonetheless, the improvement is minor (0.09%). Finally, following the findings, we choose to use CLIP as the text encoder for this paper.

Table 7. Accuracies of our NAYER with different text encoders.

	CIFAR-10				CIFAR-100			
Text Encoder	SOTA	Doc2Vec	SBERT	CLIP	SOTA	Doc2Vec	SBERT	CLIP
Accuracy	93.25	93.98	93.94	**94.07**	69.95	71.58	71.63	**71.72**

5. Conclusion

In this paper, we propose a novel DFKD method, namely NAYER, which utilizes the meaningful LTE as the input and relocates the random source from the input to the noisy layer. The significance of LTE lies in its ability to contain substantial meaningful information, enabling the fast generating images in only few steps. The use of noisy layer address the overfocus problem in using constant input information and increase significantly the diversity. Our extensive experiments on different datasets and tasks prove NAYER's superiority over other SOTA DFKD methods.

Acknowledgements

This work was supported by ARC DP23 grant DP230101176 and by the Air Force Office of Scientific Research under award number FA2386-23-1-4044.

References

[1] Kuluhan Binici, Shivam Aggarwal, Nam Trung Pham, Karianto Leman, and Tulika Mitra. Robust and resource-efficient data-free knowledge distillation by generative pseudo replay. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 6089–6096, 2022. 1

[2] Kuluhan Binici, Shivam Aggarwal, Nam Trung Pham, Karianto Leman, and Tulika Mitra. Robust and resource-efficient data-free knowledge distillation by generative pseudo replay. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 6089–6096, 2022. 2, 3, 7

[3] Kuluhan Binici, Nam Trung Pham, Tulika Mitra, and Karianto Leman. Preventing catastrophic forgetting and distribution mismatch in knowledge distillation via synthetic data. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 663–671, 2022. 7

[4] Yaohui Cai, Zhewei Yao, Zhen Dong, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. Zeroq: A novel zero shot quantization framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13169–13178, 2020. 7

[5] Yoojin Choi, Jihwan Choi, Mostafa El-Khamy, and Jungwon Lee. Data-free network quantization with adversarial knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 710–711, 2020. 7

[6] Kien Do, Thai Hung Le, Dung Nguyen, Dang Nguyen, Haripriya Harikumar, Truyen Tran, Santu Rana, and Svetha Venkatesh. Momentum adversarial distillation: Handling large distribution shifts in data-free knowledge distillation. *Advances in Neural Information Processing Systems*, 35:10055–10067, 2022. 1, 2, 3, 4, 5, 6, 7

[7] Gongfan Fang, Kanya Mo, Xinchao Wang, Jie Song, Shitao Bei, Haofei Zhang, and Mingli Song. Up to 100x faster data-free knowledge distillation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 6597–6604, 2022. 2, 3, 5, 6, 7

[8] Gongfan Fang, Jie Song, Xinchao Wang, Chengchao Shen, Xingen Wang, and Mingli Song. Contrastive model inversion for data-free knowledge distillation. *arXiv preprint arXiv:2105.08584*, 2021. 1, 2, 3, 7

[9] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018. 4

[10] Michael Fu, Van Nguyen, Chakkrit Kla Tantithamthavorn, Trung Le, and Dinh Phung. Vulexplainer: A transformer-based hierarchical distillation for explaining vulnerability types. *IEEE Transactions on Software Engineering*, 2023. 1

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 6

[12] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 1

[13] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 6

[14] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196. PMLR, 2014. 2, 3, 8

[15] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015. 6

[16] Yang Li and Tao Yang. Word embedding for understanding natural language: a survey. *Guide to big data applications*, pages 83–104, 2018. 2

[17] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35, 2023. 2

[18] Yuang Liu, Wei Zhang, and Jun Wang. Zero-shot adversarial quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1512–1521, 2021. 7

[19] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. *Advances in neural information processing systems*, 31, 2018. 3

[20] Liangchen Luo, Mark Sandler, Zi Lin, Andrey Zhmoginov, and Andrew Howard. Large-scale generative data-free distillation. *arXiv preprint arXiv:2012.05578*, 2020. 3, 4

[21] Paul Micaelli and Amos J Storkey. Zero-shot knowledge transfer via adversarial belief matching. *Advances in Neural Information Processing Systems*, 32, 2019. 1, 3, 7

[22] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013. 2

[23] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 3

[24] Gaurav Kumar Nayak, Konda Reddy Mopuri, Vaisakh Shaj, Venkatesh Babu Radhakrishnan, and Anirban Chakraborty. Zero-shot knowledge distillation in deep networks. In *International Conference on Machine Learning*, pages 4743–4751. PMLR, 2019. 1

[25] Tuan Nguyen, Van Nguyen, Trung Le, He Zhao, Quan Hung Tran, and Dinh Phung. Cycle class consistency with distributional optimal transport and knowledge distillation for unsupervised domain adaptation. In *Uncertainty in Artificial Intelligence*, pages 1519–1529. PMLR, 2022. 1

[26] Thanh Nguyen-Duc, Trung Le, He Zhao, Jianfei Cai, and Dinh Phung. Adversarial local distribution regularization for knowledge distillation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 4681–4690, 2023. 1

[27] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *International conference on machine learning*, pages 2642–2651. PMLR, 2017. 3

[28] Gaurav Patel, Konda Reddy Mopuri, and Qiang Qiu. 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*, pages 7786–7794, 2023. 1, 2, 3, 5, 6, 7

[29] Cuong Pham, Van-Anh Nguyen, Trung Le, Dinh Phung, Gustavo Carneiro, and Thanh-Toan Do. Frequency attention for knowledge distillation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2277–2286, 2024. 1

[30] Zengyu Qiu, Xinzhu Ma, Kunlin Yang, Chunya Liu, Jun Hou, Shuai Yi, and Wanli Ouyang. Better teacher better student: Dynamic prior knowledge for knowledge distillation. *arXiv preprint arXiv:2206.06067*, 2022. 1

[31] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 2, 8

[32] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019. 2, 8

[33] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019. 1

[34] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 6

[35] Minh-Tuan Tran, Trung Le, Xuan-May Le, Mehrtash Harandi, and Dinh Phung. Text-enhanced data-free approach for federated class-incremental learning. *arXiv preprint arXiv:2403.14101*, 2024. 1

[36] Zhendong Yang, Zhe Li, Xiaohu Jiang, Yuan Gong, Zehuan Yuan, Danpei Zhao, and Chun Yuan. Focal and global knowledge distillation for detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4643–4652, 2022. 1

[37] Hongxu Yin, Pavlo Molchanov, Jose M Alvarez, Zhizhong Li, Arun Mallya, Derek Hoiem, Niraj K Jha, and Jan Kautz. Dreaming to distill: Data-free knowledge transfer via deepinversion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8715–8724, 2020. 1, 2, 3, 5, 6, 7

[38] Jaemin Yoo, Minyong Cho, Taebum Kim, and U Kang. Knowledge extraction with no observable data. *Advances in Neural Information Processing Systems*, 32, 2019. 3

[39] Shikang Yu, Jiachen Chen, Hu Han, and Shuqiang Jiang. Data-free knowledge distillation via feature exchange and activation region constraint. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24266–24275, 2023. 1, 2, 3, 5, 6, 7

[40] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016. 6

[41] Borui Zhao, Quan Cui, Renjie Song, Yiyu Qiu, and Jiajun Liang. Decoupled knowledge distillation. In *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*, pages 11953–11962, 2022. 1