DISTILL VISION TRANSFORMERS TO CNNS VIA LOW-RANK REPRESENTATION APPROXIMATION

Anonymous authors
Paper under double-blind review

ABSTRACT

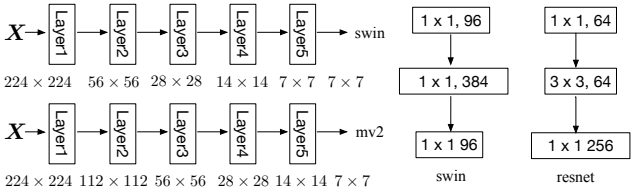
Vision Transformers attain state-of-the-art performance in diverse vision tasks thanks to their scalable and long-range dependencies modeling. Meanwhile, CNNs are still practical and efficient in many industry scenarios, thanks to their inductive biases and mature tiny architectures. Thus it is a challenging yet interesting problem to study the Knowledge Distillation (KD) of these two different architectures. In particular, how to transfer global information from Vision Transformers to tiny CNNs. We point out that many current CNN distillation methods are ineffective in the Vision Transformers distillation scenario, which implies that distilling global information is not easy due to the architecture gaps. We develop an encoder-decoder representation distillation framework, namely **L**ow **R**ank **R**epresentation **A**pproximation (LRRA), to address the problem. The key insight of LRRA is that the global information modeling can be seen as finding the most important bases and corresponding codes. This process can be solved by matrix decomposition. Specifically, the student representation is encoded to a low-rank latent representation and used to approximate the teacher representation. The most distinguishable knowledge, i.e., global information, is distilled via the low-rank representation approximation. The proposed method offers a potential closed-form solution without introducing extra learnable parameters and hand-crafted engineering. We benchmark 11 KD methods to demonstrate the usefulness of our approach. Extensive ablation studies validate the necessity of the low-rank structure.

1 INTRODUCTION

As a general foundation model architecture, transformers have achieved state-of-the-art performance in many research fields such as language, vision, and multimodal (Devlin et al., 2019; Brown et al., 2020; Liu et al., 2021; Bao et al., 2021; Wang et al., 2022; He et al., 2022), thanks to their strong modeling of long-range dependencies and the scalability of handling massive datasets. Though transformers have achieved huge success, their high complexity is the bottleneck for deploying them in industrial scenarios. The self-attention mechanism requires quadratic complexity, i.e., $\mathcal{O}(n^2d)$, with the sequence length n the dimension d, which makes both training and deployment costly when the sequence length is large.

On the contrary, Convolutional Neural Networks (CNNs) are practical and efficient in many industry scenarios. One of the most important inductive biases of CNNs is translation equivalence, which is useful in many visual tasks because of its equivariant representations. In addition, CNNs are deployment friendly as most of them consist of small kernels. There exist many mature tiny CNN backbones (e.g., MobileNet (Sandler et al., 2018), ShuffleNet (Zhang et al., 2018)) which have proven to be practical in real applications. However, CNNs are also well known for their poor modeling of global information (Wang et al., 2018; Raghu et al., 2021; Park & Kim, 2022).

From these perspectives, it is interesting to study Knowledge Distillation (KD) between these two architectures. Previous works such as Deit (Touvron et al., 2021) also show the potential of knowledge distillation between these two architectures. It adopts a simple logit-level distillation from a large CNN model to a pure ViT model such that the ViT model does not need a large labeled dataset to complement inductive bias. Inversely, our work focuses on transferring global information from vision transformers to CNNs. Except for distillation via logits, we aim to distill vision transformers



(a) different downsamples (b) different bottlenecks

Stage-ratio	Stage1	Stage2	Stage3	Stage4
ResNet18	2	2	2	2
ResNet50	3	4	6	3
Swin-base	2	2	18	2

(c) different configures

Figure 1: Three types of architecture gaps between Transformers and CNNs. (a) Transformers are practical in dealing with long-sequential data but suffer from high complexity. Thus they downsample the image into a relatively small size at the first stage compared with CNNs (e.g., mobilenetv2 (mv2) (Sandler et al., 2018).). (b) Local vision transformers such as SwinTransformer adopt a spindle-shaped bottleneck design, while some CNN architectures such as ResNet (He et al., 2016) apply a pyramid-shaped bottleneck. (c) Many advanced architectures such as ConvNext (Liu et al., 2022) and SwinTransformer (Liu et al., 2021) add more blocks in stage3.

at the representation level. One reason is that the representation contains informative knowledge as their dimensions exhibit complex interdependencies (Tian et al., 2020). Another attractive motivation is that some foundation models which are trained in a self-supervised manner (Devlin et al., 2019; Bao et al., 2021; He et al., 2022) may not contain a classification head to produce logits. Thus, representation is the most general target for distillation since it does not rely on the tasks and model structures.

Since we give more attention to results in compact CNNs, KD methods that distill knowledge to CNNs are our main competitors. Nevertheless, current KD methods could be more effective in vision transformer distillation scenarios. Subsequently, many carefully designed KD methods can not achieve comparable results to some pioneer works, such as the vanilla KD (Hinton et al., 2015), FitNet (Romero et al., 2014). We conjure the architecture gaps as the main reason that induces the negative transfer of these KD methods. As shown in Fig. 1, some classical vision transformers, such as SwinTransformer (Liu et al., 2021) show different architecture designs compared with those in CNNs. Except for self-attention and convolution, some designs, including downsampling strategy, bottleneck design, and stage configurations, are worth noticing. For example, SwinTransformer downsamples the image into a smaller size, i.e., 56×56 at the first stage. The difference in downsampling strategies may induce extra information loss because students' outputs have to be downsampled into the same size to match the teacher.

We propose an encoder-decoder representation distillation structure to address the issue. We seek to design an encoder-decoder structure to "translate" the student to teacher, thus bypassing the negative transfer due to the existing huge architecture gaps.

Since transformers contain rich global correlations, designing an attention-based projection module seems like a potential solution for CNNs to attain more global information. However, this approach may involve (1) designing a sophisticated attention-based projection module and (2) heavy hyperparameters tuning. Motivated by the insights that the attention module can be replaced by matrix decomposition (Li et al., 2019; Geng et al., 2021), from which global information modeling can be seen as finding a dictionary and corresponding code, we model knowledge distillation as a low-rank approximation problem and solve it via matrix decomposition. The motivation of representation decomposition is not trivial. The success of logit-level KD (Hinton et al., 2015) is based on the assumption that rich logit-level relationships can complement the one-hot supervised signal. Therefore we can speculate that the representation, which is right before the logits output, is also correlated to some extent, even though we hope all dimensions are independent. Then representations could be decomposed as the linear combination of bases whose dimension is much lower than the original. All in all, our approach first decomposes the student representation into a low-rank latent representation, then uses this latent representation to approximate the teacher representation. As both encoding and decoding are down by matrix decomposition, the distillation process requires no extra learnable projection module with less hand-crafted engineering.

In summary, our contributions are as follows:

- We show that many CNN KD methods are ineffective when regarding vision transformer as a teacher due to the architecture gaps between Vision Transformers and CNNs.
- We present Low-Rank Representation Approximation (LRRA), which can be solved by matrix decomposition and introduces a closed-form solution. This solution shows the potential to model global information using matrix factorization rather than attention.
- Benchmarking 11 recent distillation methods on ImageNet100 (Deng et al., 2009; Wang & Isola, 2020). Our method shows a competitive result without introducing any extra learnable parameters.

2 RELATED WORK

Knowledge Distillation in CNN. As a model compression technique (Buciluǎ et al., 2006), knowledge distillation aims to transfer the knowledge from a teacher model to a student model. In this paper, we divide KD methods into two categories.

(1) No-parametric methods. Works in this line rely on something other than extra learnable projection parameters to map the student's output to the teacher's into the same size. The seminal work KD (Hinton et al., 2015) measures the KL-divergence between teacher and student's logits output. SP (Tung & Mori, 2019) measures the pairwise Gram matrix, which aims to distill the sample correlations from teacher to student. RKD (Park et al., 2019) introduces mutual relations that attempt to transfer structure-wise relations. DKD (Zhao et al., 2022) decouples the logits-level KD into positive and negative terms and implicitly uses target information to conduct hard pair mining on knowledge distillation. One of the disadvantages of no-parametric KD methods is the information loss when dealing with different dimension sizes, where different dimensions are usually averaged, which somehow ruins the relationship and similarities.

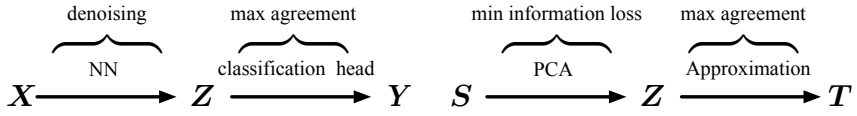
(2) Distillation methods rely on extra training (projection) parameters. These works mainly focus on distilling knowledge at the feature level. The straightforward way is using one MLP or Convolution layer to map student to teacher into the same dimension. For example, FitNet (Romero et al., 2014) leverages a convolution layer to map the student feature map and the teacher feature map into the same size. CRD (Tian et al., 2020) uses projection heads, i.e., MLP layers on both student and teacher, to map their representations into the same dimension. Often the dimensions are reduced to a lower number to reduce the computational cost due to contrastive learning. MGD (Yang et al., 2022) utilized two convolutions layers to recover teacher feature maps from the highly masked student feature map. The multiple convolution layers are served as a decoder in an auto-regression framework. Knowledge Review (Chen et al., 2021) formulates multiple layers to establish an FPN-like distillation structure. Although these extra projection parameters induce more computational complexity in the distillation framework, they are not a bad choice in most cases since the projection layers soften the distillation objective, which partially prevents negative transfer. *One important motivation in our work is to explore a closed solution rather than a learnable projection head in a distillation scenario*.

Matrix Decomposition in modeling global information. Our work is highly motivated by the recent progress in applying matrix decomposition in modeling global information. As a common model compression technique, low-rank decomposition aims to decompose the kernel, i.e., the DNN layer, into smaller kernels in both the training and inference stage. On the other hand, recent studies also show that matrix decomposition has a strong global information modeling ability, even compared with self-attention. EMANet (Li et al., 2019) introduces a matrix decomposition approach that mimics attention's modeling ability and is optimized via an alternating optimization approach. Hamburger (Geng et al., 2021) further proposes a hamburger-like structure where the key ingredient is a hand-designed matrix decomposition module to replace the attention module. These two methods show that a well-designed matrix decomposition module can behave similarly or even better than an attention module. Even though both methods still need to learn some projection parameters, like bases, and need a carefully designed initialization strategy. The key insight behind them is that matrix decomposition induces low-rank structure, which is why self-attention has such a powerful modeling ability. The softmax operation in the self-attention module induces the low-rank structure naturally. Some recent studies on analyzing the role of self-attention (Tay et al., 2021) reveal that even a random initialized matrix trained with a softmax operator can achieve a competitive result compared with the self-attention module.

3 METHODOLOGY

We focus on distilling Vision Transformers to CNNs without a painstaking hand-crafted design. We define a batch of images $\boldsymbol{X} \in \mathbb{R}^{b\times c\times h\times w}$ as the input, $\boldsymbol{S} \in \mathbb{R}^{b\times p_s}$ and $\boldsymbol{T} \in \mathbb{R}^{b\times p_t}$ as the representation of student and teacher networks, respectively. The latent representation $\boldsymbol{Z} \in \mathbb{R}^{b\times p_z}$ where the dimension size of p_z is normally a much smaller number compared with p_s and p_t. In our setting, b, c, h, w represent the batch-size, number of channels, height and weight, respectively. p is the dimension size of the representation, and both student and teacher representations are the output of the penultimate layer, which is the second last layer (i.e., before the classification head) of the networks. Our distillation framework can be understood as an encoder-decoder structure, where in encoder stage, we aim to learn a mapping function $\boldsymbol{W_1} : \boldsymbol{S} \to \boldsymbol{Z}$, and in decoder stage, we attempt to recover $\boldsymbol{Z}$ to $\boldsymbol{T}$ by a linear transformation $\boldsymbol{W_2}$. Later we will show that $\boldsymbol{W_1}$ and $\boldsymbol{W_2}$ do not need to be learnable in our framework.

Encoder-Decoder architecture design. We start by introducing our encoder-decoder architecture design. We aim to provide the intuition of encoder-decoder design and why the latent representation $\boldsymbol{Z}$ is supposed to be low-rank. As shown in Fig. 2, our encoder-decoder framework can be explained in an information bottleneck plane (Shwartz-Ziv & Tishby, 2017). Overall, the objective of information bottleneck is:



(a) Information bottleneck framework (b) Our LRRA framework

Figure 2: (a) Information bottleneck pipeline. $\boldsymbol{X}$ is the input image, $\boldsymbol{Z}$ is the latent representation, and $\boldsymbol{Y}$ is the target label. The information bottleneck framework investigates the NN training process though information plane analysis, where the NN training objective is to maximize $I(\boldsymbol{Z};\boldsymbol{Y})$ while minimizing the $I(\boldsymbol{X};\boldsymbol{Z})$, $I(\cdot)$ denotes mutual information. (b) Our low-rank representation framework follows the design rule of information bottleneck. Therefore we aim to learn a latent representation $\boldsymbol{Z}$ that satisfies the minimal sufficient statistics, which is achieved by maximizing the agreement of $\boldsymbol{Z}$ and $\boldsymbol{T}$ while extracting the most informative feature from $\boldsymbol{S}$.

$$\min\{I(\boldsymbol{X};\boldsymbol{Z}) - \beta I(\boldsymbol{Z};\boldsymbol{Y})\} \tag{1}$$

This objective (1) can be understood from image denoising or feature extracting perspective, in which the most compact and sufficient features $\boldsymbol{Z}$ are learned by the tradeoff between compression of input $\boldsymbol{X}$ and prediction of the target $\boldsymbol{Y}$. The compression is often obtained by an NN model, e.g., CNN's feature extraction. In the distillation scenario, our goal is to map $\boldsymbol{S}$ to a latent representation $\boldsymbol{Z}$ with minimal information loss and maximize the agreement between $\boldsymbol{Z}$ and $\boldsymbol{T}$. We often wish the latent representation $\boldsymbol{Z}$ is as compact as possible. Thus the most valuable information is maintained (Yu et al., 2020). The information bottleneck plane quantifies the relevant information via mutual information metric, which has some useful and important properties (Shwartz-Ziv & Tishby, 2017):

Theorem 1. *For any latent variables $\boldsymbol{S}$ and $\boldsymbol{T}$, their mutual information is invariance to any invertible functions Φ and Ψ:*

$$I(\boldsymbol{S};\boldsymbol{T}) = I(\Phi(\boldsymbol{S});\Psi(\boldsymbol{T})). \tag{2}$$

Theorem 1 show that if the whole system is invertible, then there's no mutual information loss.

Theorem 2. *Any representation variable, $\boldsymbol{Z}$, characterized by encoder-decoder distributions, $P(\boldsymbol{Z}|\boldsymbol{S})$ and $P(\boldsymbol{T}|\boldsymbol{Z})$, then the information path satisfies the following Data Processing Inequality (DPI) chains:*

$$I(\boldsymbol{S};\boldsymbol{T}) \geq I(\boldsymbol{Z};\boldsymbol{T}). \tag{3}$$

Theorem 2 implies that we can always achieve a better bound if we can guarantee the information loss is constrained in the encoder part, i.e., $\boldsymbol{S} \to \boldsymbol{Z}$.

Our distillation framework follows the design rule of the information bottleneck plane. The whole pipeline is almost invertible, as we will show later. In encoder part, we use PCA (Pearson,

1901; Wold et al., 1987) to compress student representation $\boldsymbol{S}$ into a low-rank latent representation $\boldsymbol{Z}$ thus the most distinguishable principal components are preserved with minimal information loss (Linsker, 1988; Deco & Obradovic, 1996). In the decoder part, we obtain a closed solution by some weak conditions to maximize the agreement between $\boldsymbol{Z}$ and $\boldsymbol{T}$, which we will give more details about later.

Encoder. In the encoder part, we can use different matrix decomposition algorithms such as VQ (Gray & Neuhoff, 1998), CD (Dhillon & Modha, 2001) or NMF (Lee & Seung, 1999). In this paper, we simply choose PCA (Pearson, 1901; Wold et al., 1987), an algorithm invented over one hundred years, for its simplicity and concise. We can formulate the encoder as:

$$\hat{\boldsymbol{W}}_{\mathbf{1}} = \arg\min_{\boldsymbol{W}_{\mathbf{1}}} \left\| \boldsymbol{S} - \boldsymbol{S}\boldsymbol{W}_{\mathbf{1}}\boldsymbol{W}_{\mathbf{1}}^{\top} \right\|^2 \quad \text{s.t.} \boldsymbol{W}_{\mathbf{1}}^{\top}\boldsymbol{W}_{\mathbf{1}} = \boldsymbol{I}_{p_z}, \tag{4}$$

where p_z is the selected PCA components, and $\boldsymbol{W}_1 \in \mathbb{R}^{p_s \times p_z}$ is the orthogonal projection matrix. We can obtain the closed solution $\boldsymbol{W}_1 = \boldsymbol{V}$ where $\boldsymbol{V}$ is eigen-decompostion of $\boldsymbol{S}^{\top}\boldsymbol{S}$. The optimal z dimensional plane is z principal components. One problem in PCA is that the dimensions are larger than batch size in most cases. To address this issue, we adopt a memory bank strategy as in (Wu et al., 2018; He et al., 2020). Then we can guarantee that the number of samples is always larger than the size of the dimensions.

Decoder. In the decoder part, we aim to maximize the agreement between latent representation $\boldsymbol{Z}$ and teacher representation $\boldsymbol{T}$. In the simplest case, we attempt to learn a linear transformation $\boldsymbol{W}_2 \in \mathbb{R}^{p_s \times p_t}$ that obtains: $\boldsymbol{T} = \boldsymbol{Z}\boldsymbol{W}_2$. Then we can formulate our objective as $\arg\min_{\boldsymbol{W}_2} \|\boldsymbol{T} - \boldsymbol{Z}\boldsymbol{W}_2\|_F^2$. Using a linear transformation to map representations of the student and teacher into the same size is commonly seen in the distillation scenario (Romero et al., 2014). On the other hand, in Self Supervised Learning (SSL) setting, we often need a $\boldsymbol{W}$, that is, a projection head (Chen et al., 2020; Grill et al., 2020; Chen & He, 2021) as well, that mainly acts as a buffer zone to prevent optimizing to trivial solutions. Recent study (Tian et al., 2021) shows that a projection head does not need to be learnable, which highly motivates our study.

By assuming some constraints, we attempt to obtain a closed solution of $\boldsymbol{W}_2$. Often we hope $\boldsymbol{W}_2$ is invertible because it guarantees the optimization stability and ensures the inverse mapping (Cortes et al., 2012). However, it is not easy to make $\boldsymbol{W}_2$ invertible, even though to form it into a simple linear transformation. We try to avoid using complicated solutions such as normalizing flow. Instead, we consider orthogonality as a desirable constraint for $\boldsymbol{W}_2$. Now we assume $\boldsymbol{Z}$ has the same size as $\boldsymbol{T}$ but with $(p_t - p_z)$ zero padding. Then $\boldsymbol{W}_2$ is a full orthogonal matrix.

Lemma 3. *All orthogonal matrices are invertible. If $\boldsymbol{W}$ is an orthogonal matrix, then:*

$$\boldsymbol{W}^{\top} = \boldsymbol{W}^{-1}. \tag{5}$$

Lemma 4. *All orthogonal matrices preserve dot product. For any random variables, $\mathbf{u}$ and $\mathbf{v}$ are in the same dimension. If $\boldsymbol{W}$ is an orthogonal matrix, then:*

$$\mathbf{u}^{\top}\mathbf{v} = (\boldsymbol{W}\mathbf{u})^{\top}(\boldsymbol{W}\mathbf{v}). \tag{6}$$

Lemma 5. *All orthogonal matrices preserve vector lengths. For any random variables $\mathbf{x}$. If $\boldsymbol{W}$ is an orthogonal matrix, then:*

$$\|\boldsymbol{W}\mathbf{x}\|^2 = \|\mathbf{x}\|^2. \tag{7}$$

By Lemma 3, we can ensure that all orthogonal matrices are invertible, which is crucial, as discussed previously. Lemma 4 and Lemma 5 are even more important because they show that geometry structure is preserved with orthogonal transformation. Now we can formulate the problem as finding an orthogonal transformation that produces the smallest error:

$$\hat{\boldsymbol{W}}_{\mathbf{2}} = \arg\min_{\boldsymbol{W}_{\mathbf{2}}} \|\boldsymbol{T} - \boldsymbol{Z}\boldsymbol{W}_2\|_F^2 \quad \text{s.t.} \boldsymbol{W}_{\mathbf{2}}^{\top}\boldsymbol{W}_{\mathbf{2}} = \boldsymbol{I}. \tag{8}$$

This problem is the classical Orthogonal Procrustes Problem and the corresponding solution can be traced back to (Schönemann, 1966). We can obtain the solution by:

$$\begin{aligned} \hat{\boldsymbol{W}}_{\mathbf{2}} &= \arg\min_{\boldsymbol{W}_{\mathbf{2}}} \|\boldsymbol{T} - \boldsymbol{Z}\boldsymbol{W}_2\|_F^2 \quad \text{s.t.} \quad \boldsymbol{W}_{\mathbf{2}}^{\top}\boldsymbol{W}_{\mathbf{2}} = I. \\ &= \arg\min_{\boldsymbol{W}_{\mathbf{2}}} \operatorname{tr}(\boldsymbol{T} - \boldsymbol{Z}\boldsymbol{W}_2)^{\top}(\boldsymbol{T} - \boldsymbol{Z}\boldsymbol{W}_2) \\ &= \arg\min_{\boldsymbol{W}_{\mathbf{2}}} \|\boldsymbol{T}\|_F^2 + \|\boldsymbol{Z}\|_F^2 - 2\operatorname{tr}(\boldsymbol{T}^{\top}\boldsymbol{Z}\boldsymbol{W}_2)). \end{aligned} \tag{9}$$

Practically, we adopt an alternating optimization strategy to obtain the solution, which means in step one, we fix $\boldsymbol{T}$ and $\boldsymbol{S}$ and obtain $\boldsymbol{W}_2$, in step two, we fix $\boldsymbol{W}_2$ and update $\boldsymbol{T}$ and $\boldsymbol{S}$. Overall, we only need to consider $\boldsymbol{S}$ in the distillation setting. It is naturally suitable for the neural network mini-batch training paradigm.

Then we only need to take $\operatorname{tr}(\boldsymbol{T}^\top \boldsymbol{Z}\boldsymbol{W}_2))$ into consideration. The closed solution is $\boldsymbol{W}_2 = \boldsymbol{U}\boldsymbol{V}^\top$ where $\boldsymbol{U}$ and $\boldsymbol{V}$ are the Singular Value Decomposition (SVD) of $\boldsymbol{Z}^\top\boldsymbol{T}$. Then we can define the decoder loss as:

$$\mathcal{L}_{decoder} = -\operatorname{tr}(\boldsymbol{T}^\top \boldsymbol{Z}\boldsymbol{W}_2) = -\operatorname{tr}(\boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{U}^\top \boldsymbol{W}_2) = -\operatorname{tr}(\boldsymbol{\Sigma}). \tag{10}$$

Discussion on decoder loss. Here we give more details and analysis on decoder loss. In the forward step, we get both teacher and student representations, i.e., $\boldsymbol{T}$ and $\boldsymbol{S}$, then $\boldsymbol{Z}$ is encoded by $\boldsymbol{S}$. $\boldsymbol{W}_2 = \boldsymbol{U}\boldsymbol{V}^\top$ where $\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top = \boldsymbol{Z}\boldsymbol{T}^\top$. All the above can be treated as the forward process, and nothing is learnable except the parameters of the student. In the backward step, since $\boldsymbol{U}$ is eliminated as introduced, we only need to consider $\boldsymbol{\Sigma}$ and our objective is to maximize the $\boldsymbol{\Sigma}$. We use $-\boldsymbol{\Sigma}$ as decode loss and only optimize $\boldsymbol{S}$, i.e., the CNN parameters of the student.

Assume $\boldsymbol{T}$ and $\boldsymbol{Z}$ have the same size, we can observe that:

$$\begin{aligned}
\mathcal{L}_{decoder} &= \|\boldsymbol{T}\|_F^2 + \|\boldsymbol{Z}\|_F^2 - 2\operatorname{tr}(\boldsymbol{\Sigma}) \\
&\leq \|\boldsymbol{T}\|_F^2 + \|\boldsymbol{Z}\|_F^2 - 2\operatorname{tr}(\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top) \\
&= \|\boldsymbol{T}\|_F^2 + \|\boldsymbol{Z}\|_F^2 - 2\operatorname{tr}(\boldsymbol{Z}^\top\boldsymbol{T}) = \|\boldsymbol{T} - \boldsymbol{Z}\|_F^2 .
\end{aligned} \tag{11}$$

Then we can find that the proposed decoder loss is always a lower bound of MSE loss between $\boldsymbol{T}$ and $\boldsymbol{Z}$. It also implies that one MLP layer projection head may have a closed solution.

We also notice that previous NLP works (Hamilton et al., 2016; Smith et al., 2017) proposed a similar solution to align word embeddings from different domains. It is a very interesting point as we can regard different features of the same image extracted by CNNs and Vision Transformers as a positive pair of features from different domains.

Our training loss is the combination of classification loss and distillation loss:

$$\mathcal{L}_{total} = \mathcal{L}_{cls} + \lambda\mathcal{L}_{dis}, \tag{12}$$

where $\mathcal{L}_{cls}$ is the cross entropy loss and λ is the trade-off parameters.

4 EXPERIMENTS

We evaluated LRRA by measuring classification accuracy on ImageNet100 with 11 KD methods. We mainly focus on tiny efficient CNN backbones. Thus we ignore some new CNNs such as ConvNext and RepLKNet (Liu et al., 2022; Ding et al., 2022), which require some transformer training tricks, as well as some deployment unfriendly modules such as large kernels. We experimented with six commonly used tiny CNN architectures including MobileNetV2 (Sandler et al., 2018), MobileNetV3 (Howard et al., 2019), VGG8 (Simonyan & Zisserman, 2014), ResNet18 (He et al., 2016), ShuffleNetV1 (Zhang et al., 2018) and ShuffleNetV2 (Ma et al., 2018). We use SwinTransformer-Base (Liu et al., 2021) as our teacher model. We also provide ablation studies on the proposed encoder-decoder design, showing that both encoder and decoder designs are necessary.

Datasets. We validate our algorithm mainly on two datasets. **ImageNet1k** (Deng et al., 2009) is a widely used image classification dataset which contains over 1280k images with 1000 categories. **ImageNet100** is a subset of ImageNet which contains roughly 120k images. We followed the same splitting rules used in (Wang & Isola, 2020). To our knowledge, few works study distilling vision transformers to CNNs. Thus in this paper, we attempt to establish a new benchmark for this setting with a middle-size dataset. Especially in the vision transformers distillation scenario, Cifar100 seems like an improper choice since its image size is only 32×32.

Implementation details. Our implementation is mainly bulit on ConvNext (Liu et al., 2022) and CRD (Tian et al., 2020). All methods follow the same training protocol with AdamW (Loshchilov & Hutter, 2017) optimizer, warmup, and cosine decay learning strategy (Loshchilov & Hutter, 2016). Advanced augmentations such as mixup (Zhang et al., 2017), cutmix (Yun et al., 2019),

Table 1: Top-1 accuracies of teacher and student networks on ImageNet100 of 11 KD methods. We follow most implementations introduced in CRD (Tian et al., 2020) and use the open-source code of other recent KD methods such as DKD (Zhao et al., 2022), MGD (Yang et al., 2022) and Review (Chen et al., 2021). Since some methods are designed for the CNN KD scenario, we slightly modified some hyper-parameters to obtain better results. Please refer to the appendix for more training details and citations of other KD methods.

Teacher Student	Swin MobileNetV2	Swin MobileNetV3	Swin VGG8	Swin ResNet18	Swin ShuffleNetV1	Swin ShuffleNetV2	Avg. Avg.
Teacher	94.48%	94.48%	94.48%	94.48%	94.48%	94.48%	94.48%
Student	84.70%	86.44%	78.86%	85.24%	77.30%	79.52%	82.01%
KD	85.34%	86.82%	79.04%	85.54%	77.46%	79.86%	82.34%
DKD	84.90%	86.86%	78.70%	85.96%	78.30%	80.02%	82.45%
AT	82.92%	66.86%	76.90%	84.80%	74.92%	76.74%	77.19%
SP	83.82%	85.26%	74.30%	84.66%	74.16%	76.12%	79.71%
RKD	78.68%	85.06%	76.52%	85.24%	75.42%	77.48%	79.73%
PKT	84.32%	86.84%	76.82%	85.20%	76.96%	78.86%	81.50%
FitNet	84.70%	86.44%	78.86%	85.24%	77.30%	79.52%	82.01%
VID	85.02%	86.46%	78.94%	86.10%	77.92%	80.06%	82.42%
CRD†	85.44%	86.18%	79.16%	85.78%	74.72%	77.88%	81.52%
Review	84.36%	86.96%	78.90%	**86.74%**	**78.62%**	79.80%	82.56%
MGD	84.52%	85.48%	78.72%	85.62%	77.98%	79.62%	81.99%
LRRA	**85.94%**	87.33%	78.76%	86.12%	77.70%	79.90%	82.62%
LRRA+DKD	85.74%	**87.42%**	**79.48%**	86.18%	78.12%	**80.14%**	**82.85%**

Table 2: Training parameters comparison between different KD methods. We mainly compare our approach with KD methods with extra training projection parameters because we observe these methods gain a better result than the no-parametric methods. For all of these methods, Review (Chen et al., 2021) obtains the most competitive results. However, it also costs almost triple times of the training projection parameters, namely the ABF module in their paper. In comparison, our method does not require any learnable projection parameters.

Teacher Student	Swin MobileNetV2	Swin MobileNetV3	Swin VGG8	Swin ResNet18	Swin ShuffleNetV1	Swin ShuffleNetV2	Avg. Avg.
Teacher	86.85M	86.85M	86.85M	86.85M	86.85M	86.85M	86.85M
Student	2.35M	4.33M	3.97M	11.22M	0.95M	1.36M	3.80M
FitNet	+0.15M	+0.19M	+1.18M	+0.59M	+0.25M	+0.12M	+0.41M
VID	+6.28M	+5.98M	+6.06M	+5.72M	+6.32M	+5.49M	+5.98M
CRD	+0.30M	+0.25M	+0.20M	+0.20M	+0.50M	+0.26M	+0.24M
Review	+9.56M	+9.40M	+9.35M	+9.24M	+9.54M	+9.34M	+9.40M
MGD	+0.31M	+0.32M	+0.44M	+0.33M	+0.57M	+0.42M	+0.40M
LRRA	✗	✗	✗	✗	✗	✗	✗

Table 3: Top-1 and Top-5 accuracies of student network ResNet18 on ImageNet validation set. We use SwinTransformer-Base, released by the official repository, as the teacher network. For all KD methods, we adopt the same training protocol.

	Teacher	Student	Student†	KD	CRD	MGD	Review	LRRA	**LRRA+DKD**
Top-1	85.17%	69.76%	70.80%	70.68%	71.38%	70.93%	71.14%	71.46%	**71.92%**
Top-5	97.48%	89.08%	89.74%	90.23%	90.24%	89.78%	90.21%	90.14%	**90.46%**

autoaug (Cubuk et al., 2018) are applied based on timm (Wightman, 2019). All Imagenet100 benchmarks are trained on 120 epochs. We adopted a strong training strategy to validate the effectiveness of KD methods based on a stronger baseline. We observed that some methods, such as (Tian et al., 2020; Chen et al., 2021), are unstable in our distillation setting. Thus we apply some modifications to these methods, which we will discuss in the appendix. For most KD methods, we use the same training hyperparameters introduced in (Tian et al., 2020) or corresponding open-source codebases. For ImageNet1k, we directly use official pretrained weights of SwinTransformer (Liu et al., 2021).

Table 4: Ablation study on different projection dimensions on the encoder.

	Student	128	256	512	1024	2048	4096	8192	12288	Ours
MV2	84.7%	85.34%	85.18%	85.04%	85.10%	85.28%	85.34%	85.40%	85.38%	**85.94%**

Table 5: Ablation study on random masks on the encoder.

	MobileV2	MobileV3	VGG8	ResNet18	ShuffleV1	ShuffleV2
Student-Acc	84.70%	86.44%	78.86%	85.24%	77.30%	79.52%
Mask-Acc	84.98%	86.70%	78.42%	85.28%	76.92%	78.74%
Ours-Acc	85.94%	87.33%	78.76%	85.88%	77.70%	79.90%

Table 6: Ablation study on the usefulness of encoder.

	MobileV2	MobileV3	VGG8	ResNet18	ShuffleV1	ShuffleV2
Student-Acc	84.70%	86.44%	78.86%	85.24%	77.30%	79.52%
Decoder-Acc	84.82%	86.44%	78.66%	85.56%	76.84%	79.44%
Ours-Acc	85.94%	87.33%	78.76%	85.88%	77.70%	79.90%

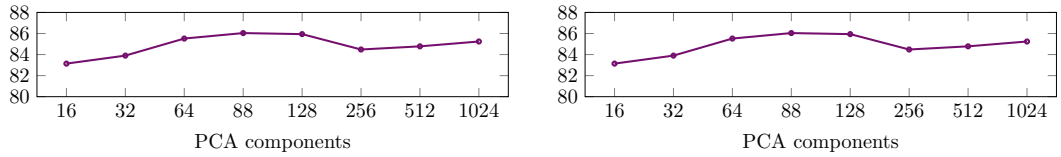
Performance. Table 1 and Table 2 compare the top-1 accuracies and extra projection parameters of different KD methods. For no-parametric KD methods, we notice that all methods' results are suffered from the architectures gap except logits-level KD methods, including DKD (Zhao et al., 2022) and KD (Hinton et al., 2015). Among them, AT Zagoruyko & Komodakis (2016) is an attention-based method that measures the similarities between features. SP Tung & Mori (2019) measures the pairwise Gram matrix of the middle feature outputs. RKD Park et al. (2019) aligns the features via mutual relations. They all suffer from the architecture gaps. For parametric KD methods, some methods with heavy projection parameters, such as VID (Ahn et al., 2019) and Review (Chen et al., 2021), achieve a competitive result, nevertheless, at the cost of almost double or even triple extra trainable parameters. We also notice that Review (Chen et al., 2021) suffers from an unstable training process even though it finally obtained a pretty good result. Overall, our method achieves the best performance on average without any extra training parameters. Our method is also complement to other logits-level method, such as DKD (Zhao et al., 2022). The combination of LRRA and DKD attain a even better result. Table 4 compares the top-1 and top-5 accuracies on ImageNet1k. Our method still achieves the best performance compared with other methods. † means the self-implementation results.

Ablation studies. We conduct ablation studies both on the encoder and decoder. In the encoder, we are interested in exploring the necessity and sensitivity of PCA with the following questions.

Firstly, we want to explore the effectiveness of PCA. Is it really useful, or a simple MLP layer can replace it? A learnable MLP layer is almost invertible and flexible, which plays an important role in self-supervised learning paradigm (Grill et al., 2020; Chen & He, 2021). As shown in Table 4, we substitute PCA with an MLP layer. Interestingly, even though the projection size was up to 12288, it did not improve the distillation result too much. The result implies that one MLP projection layer may have a closed-form solution, which can be obtained by matrix decomposition.

On the other hand, LRRA introduces PCA to extract the most distinguishable features from original representations. Can any forms of low-rank representation be used in distillation? To verify the hypothesis, we randomly mask student representation into the same size as default PCA components. The random mask module is inspired by (Yang et al., 2022) from which the masked pixels in representation are directly set to zero. As shown in Table 5, a random mask may not be an effective strategy in distillation, demonstrating the usefulness of PCA.

Do we need the encoder in our framework? We make an ablation study to validate the usefulness of the encoder. As we can observe in Table 6, we abandon the encoder and directly use the decoder for distillation. The whole distillation performance drops a lot without an encoder. We conjure a potential reason is the curse of dimensionality. Since both student and teacher representations are in high dimension, making the optimization process become more complex.



(a) different pca components on MobileNetV2

(b) different pca components on ShuffleNetV1

Figure 3: Ablation study on different PCA components on encoder. (a) different pca components on MobileNetV2. (b) different pca components on ShuffleNetV1.

Table 7: Ablation study on decoder

	MobileV2	MobileV3	VGG8	ResNet18	ShuffleV1	ShuffleV2
Student-Acc	84.70%	86.44%	78.86%	85.24%	77.30%	79.52%
Proj-Acc	84.80%	86.46%	78.66%	85.18%	77.10%	79.32%
Ours-Acc	85.94%	87.33%	78.76%	85.88%	77.70%	79.90%

The choice of PCA components is a hyper-parameter in our distillation framework, and we also show ablation results here. As we can see in Figure 3, large components do not contribute to the performance. On the contrary, some medium or even small components, such as 64 or 88, achieve strong performance. However, if we lower the components to a very small number, i.e., 32, it did not perform well.

Finally, we want to explore the role of the decoder. Instead of using the previous closed-form solution, we replace the decoder with a learnable projection to map the latent representation to the teacher representation of the same size. As shown in Table 7, unlike using a projection on the encoder, a projection on the decoder somehow hinders the performance, which shows the usefulness of our decoder design.

5 Discussion and Conclusion

Currently, our study mainly focuses on representation-level distillation. Although representation-level distillation is a more general solution, multi-stage feature distillation may perform better in some cases. The combination of multi-stage features and low-rank structure distillation may bring additional benefits. Meanwhile, a better matrix decomposition approach may exist in both encoder and decoder parts. For instance, in the encoder part, we could use LDA Fisher (1936), another classical algorithm that leverages class information to reduce dimension, which may achieve a better reduction result. Some "advanced" matrix decomposition algorithms such as Robust PCA Candès et al. (2011) or NMF Lee & Seung (1999) all could possible achieve a better result. Another interesting direction is to combine matrix decomposition with a learnable projection module as done in Li et al. (2019); Geng et al. (2021).

This paper studies distilling knowledge from vision transformers to CNNs at the representation level. We show that there exists an architecture gap between two different architectures such that many current KD methods are ineffective in a vision transformers distillation scenario. Motivated by the recent progress in applying matrix decomposition in modeling global information in vision tasks, we formulate global information distillation as a low-rank approximation problem. The low-frequency knowledge, i.e., global context knowledge of transformers, could be distilled in a low-rank structure. Then we develop an encoder-decoder distillation framework which can be solved by matrix decomposition. In the encoder stage, we compress the student representation into a low-rank latent representation via PCA. In the decoder stage, we obtain a closed-form decomposition solution by assuming transformation matrix is orthogonal. The whole distillation framework is simple and neat without extra training parameters. We benchmark 11 KD methods on ImageNet100 to demonstrate the effectiveness of our approach. We hope this work can provide some interesting findings in distillation from Vision Transformers to CNNs.

REFERENCES

Sungsoo Ahn, Shell Xu Hu, Andreas Damianou, Neil D Lawrence, and Zhenwen Dai. Variational information distillation for knowledge transfer. In *CVPR*, pp. 9163–9171, 2019.

Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *NeurIPS*, 33:1877–1901, 2020.

Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *SIGKDD*, pp. 535–541, 2006.

Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):1–37, 2011.

Pengguang Chen, Shu Liu, Hengshuang Zhao, and Jiaya Jia. Distilling knowledge via knowledge review. In *CVPR*, pp. 5008–5017, 2021.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, pp. 1597–1607. PMLR, 2020.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *CVPR*, pp. 15750–15758, 2021.

Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh. Algorithms for learning kernels based on centered alignment. *The Journal of Machine Learning Research*, 13:795–828, 2012.

Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018.

Gustavo Deco and Dragan Obradovic. *An information-theoretic approach to neural computing*. Springer Science & Business Media, 1996.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pp. 248–255. Ieee, 2009.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *NAACL-HLT*, pp. 4171–4186. Association for Computational Linguistics, 2019. doi: 10.18653/v1/n19-1423. URL https://doi.org/10.18653/v1/n19-1423.

Inderjit S Dhillon and Dharmendra S Modha. Concept decompositions for large sparse text data using clustering. *Machine learning*, 42(1):143–175, 2001.

Xiaohan Ding, Xiangyu Zhang, Jungong Han, and Guiguang Ding. Scaling up your kernels to 31x31: Revisiting large kernel design in cnns. In *CVPR*, pp. 11963–11975, 2022.

Ronald A Fisher. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7 (2):179–188, 1936.

Zhengyang Geng, Meng-Hao Guo, Hongxu Chen, Xia Li, Ke Wei, and Zhouchen Lin. Is attention better than matrix decomposition? *arXiv preprint arXiv:2109.04553*, 2021.

Robert M. Gray and David L. Neuhoff. Quantization. *IEEE transactions on information theory*, 44 (6):2325–2383, 1998.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020.

William L Hamilton, Jure Leskovec, and Dan Jurafsky. Diachronic word embeddings reveal statistical laws of semantic change. *arXiv preprint arXiv:1605.09096*, 2016.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pp. 770–778, 2016.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. *arXiv preprint arXiv:1911.05722*, 2019.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, pp. 9729–9738, 2020.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, pp. 16000–16009, 2022.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *NeurIPS*, 2015.

Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *ICCV*, pp. 1314–1324, 2019.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *ICML*, pp. 3519–3529. PMLR, 2019.

Daniel D Lee and H Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

Xia Li, Zhisheng Zhong, Jianlong Wu, Yibo Yang, Zhouchen Lin, and Hong Liu. Expectation-maximization attention networks for semantic segmentation. In *ICCV*, pp. 9167–9176, 2019.

Ralph Linsker. Self-organization in a perceptual network. *Computer*, 21(3):105–117, 1988.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, pp. 10012–10022, 2021.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *CVPR*, pp. 11976–11986, 2022.

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Wenjie Luo, Yujia Li, Raquel Urtasun, and Richard Zemel. Understanding the effective receptive field in deep convolutional neural networks. *NeurIPS*, 29, 2016.

Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *ECCV*, pp. 116–131, 2018.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Namuk Park and Songkuk Kim. How do vision transformers work? *arXiv preprint arXiv:2202.06709*, 2022.

Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *CVPR*, pp. 3967–3976, 2019.

Nikolaos Passalis and Anastasios Tefas. Learning deep representations with probabilistic knowledge transfer. In *ECCV*, pp. 268–284, 2018.

Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, 2(11):559–572, 1901.

Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do vision transformers see like convolutional neural networks? *NeurIPS*, 34:12116–12128, 2021.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*, 2014.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, pp. 4510–4520, 2018.

Peter H Schönemann. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31(1):1–10, 1966.

Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information. *arXiv preprint arXiv:1703.00810*, 2017.

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

Samuel L Smith, David HP Turban, Steven Hamblin, and Nils Y Hammerla. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. *arXiv preprint arXiv:1702.03859*, 2017.

Yi Tay, Dara Bahri, Donald Metzler, Da-Cheng Juan, Zhe Zhao, and Che Zheng. Synthesizer: Rethinking self-attention for transformer models. In *ICML*, pp. 10183–10192. PMLR, 2021.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. In *ICLR*, 2020.

Yuandong Tian, Xinlei Chen, and Surya Ganguli. Understanding self-supervised learning dynamics without contrastive pairs. In *ICML*, pp. 10268–10278. PMLR, 2021.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *ICML*, pp. 10347–10357. PMLR, 2021.

Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. In *ICCV*, pp. 1365–1374, 2019.

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *ICML*, pp. 9929–9939. PMLR, 2020.

Wenhui Wang, Hangbo Bao, Li Dong, Johan Bjorck, Zhiliang Peng, Qiang Liu, Kriti Aggarwal, Owais Khan Mohammed, Saksham Singhal, Subhojit Som, et al. Image as a foreign language: Beit pretraining for all vision and vision-language tasks. *arXiv preprint arXiv:2208.10442*, 2022.

Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, pp. 7794–7803, 2018.

Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019.

Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.

Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *CVPR*, pp. 3733–3742, 2018.

Zhendong Yang, Zhe Li, Mingqi Shao, Dachuan Shi, Zehuan Yuan, and Chun Yuan. Masked generative distillation. *arXiv preprint arXiv:2205.01529*, 2022.

Yaodong Yu, Kwan Ho Ryan Chan, Chong You, Chaobing Song, and Yi Ma. Learning diverse and discriminative representations via the principle of maximal coding rate reduction. *NeurIPS*, 33: 9422–9434, 2020.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*, pp. 6023–6032, 2019.

Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *arXiv preprint arXiv:1612.03928*, 2016.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *CVPR*, pp. 6848–6856, 2018.

Borui Zhao, Quan Cui, Renjie Song, Yiyu Qiu, and Jiajun Liang. Decoupled knowledge distillation. In *CVPR*, pp. 11953–11962, 2022.

6 APPENDIX

KD Methods Our approach compares to the following current state-of-the-art methods from the literature:

1. Knowledge Distillation (KD) (Hinton et al., 2015)
2. Decoupled Knowledge Distillation (DKD) (Zhao et al., 2022)
3. Attention Transfer (AT) (Zagoruyko & Komodakis, 2016)
4. Similarity-Preserving Knowledge Distillation (SP) (Tung & Mori, 2019)
5. Relational Knowledge Distillation (RKD) (Park et al., 2019)
6. Learning deep representations with probabilistic knowledge transfer (PKT) (Passalis & Tefas, 2018)
7. FitNets: Hints for thin deep nets (Romero et al., 2014)
8. Variational information distillation for knowledge transfer (VID) (Ahn et al., 2019)
9. Contrastive Representation Distillation (CRD) (Tian et al., 2020)
10. Distilling knowledge via knowledge review (Review) (Chen et al., 2021)
11. Masked Generative Distillation (MGD) (Yang et al., 2022)

Table of Notion

Table 8: Summary of notations in this paper

β	A scalar.
$\mathbf{x}$	A vector.
$\boldsymbol{X}$	A matrix.
$I(\cdot)$	Mutula information
$\|\cdot\|_F$	Frobenius norm

Proof of Lemma 3: The inverse of an orthogonal matrix is its transpose.

Proof. Let $\boldsymbol{W}$ be an n$\times$ n matrix:

$$\boldsymbol{W} = [\mathbf{w}_1, \mathbf{w}_2, \cdots \mathbf{w}_n], \tag{13}$$

where $\mathbf{w}_1$ is the i-th column vector, and we know that $\boldsymbol{W}^\top \boldsymbol{W} = \boldsymbol{I}$ where $\boldsymbol{I}$ is the identity matrix. Then we have:

$$(\boldsymbol{W}^\top \boldsymbol{W})_{i,j} = \mathbf{w}_i^\top \mathbf{w}_j = \begin{cases} 1, & \text{i=j;} \\ 0, & \text{i}\neq\text{j.} \end{cases} \tag{14}$$

Since the column vectors are orthogonal, then column vectors are linearly independent. We can conclude that $\boldsymbol{W}$ is invertible. Also $\boldsymbol{W}^\top \boldsymbol{W}$ is the exact description of the identity matrix. The definition of $\boldsymbol{W}^{-1}$ is $\boldsymbol{W}^{-1}\boldsymbol{W} = \boldsymbol{I}$. We have:

$$\begin{aligned} (\boldsymbol{W}^\top \boldsymbol{W})\boldsymbol{W}^{-1} &= \boldsymbol{W}^{-1} \\ &= \boldsymbol{W}^\top (\boldsymbol{W}\boldsymbol{W}^{-1}) \\ &= \boldsymbol{W}^\top. \end{aligned} \tag{15}$$

Then we prove Lemma 3. □

Proof of Lemma 4: Orthogonal projection preserves the dot product

Proof.

$$\begin{aligned} \mathbf{u}^\top \mathbf{v} &= (\boldsymbol{W}\mathbf{u})^\top (\boldsymbol{W}\mathbf{v}) \\ &= \mathbf{u}^\top \boldsymbol{W}^\top \boldsymbol{W}\mathbf{v} \\ &= \mathbf{u}^\top \mathbf{v}. \end{aligned} \tag{16}$$

Then we prove the Lemma 4. □

Proof of Lemma 5: Orthogonal projection preserves vector length

Proof.

$$\begin{aligned} \|\boldsymbol{W}\mathbf{x}\|^2 &= (\boldsymbol{W}\mathbf{x})^\top \boldsymbol{W}\mathbf{x} \\ &= \mathbf{x}^\top \boldsymbol{W}^\top \boldsymbol{W}\mathbf{x} \\ &= \mathbf{x}^\top (\boldsymbol{W}^\top \boldsymbol{W})\mathbf{x} \\ &= \mathbf{x}^\top \mathbf{x} \\ &= \mathbf{x} \cdot \mathbf{x} \\ &= \|\mathbf{x}\|^2 \end{aligned} \tag{17}$$

Then we prove Lemma 5. □

Proof of Equation (9)

Proof. We give the proof of closed-form solution of $\boldsymbol{W}_2$ in decoder. We have:

$$\begin{aligned} \hat{\boldsymbol{W}}_{\boldsymbol{2}} &= \arg\min_{\boldsymbol{W_2}} \|\boldsymbol{T} - \boldsymbol{Z}\boldsymbol{W}_2\|_F^2 \\ &= \arg\min_{\boldsymbol{W_2}} \|\boldsymbol{T} - \boldsymbol{Z}\boldsymbol{W}_2\|_F^2 \\ &= \arg\min_{\boldsymbol{W_2}} \operatorname{tr}(\boldsymbol{T} - \boldsymbol{Z}\boldsymbol{W}_2)^\top (\boldsymbol{T} - \boldsymbol{Z}\boldsymbol{W}_2) \\ &= \arg\min_{\boldsymbol{W_2}} \operatorname{tr}(\boldsymbol{T}^\top \boldsymbol{T}) + \operatorname{tr}(\boldsymbol{W}_2^\top \boldsymbol{Z}^\top \boldsymbol{Z}\boldsymbol{W}_2) - 2\operatorname{tr}(\boldsymbol{T}^\top \boldsymbol{Z}\boldsymbol{W}_2)), \end{aligned} \tag{18}$$

since $\boldsymbol{W}_2^\top \boldsymbol{W} = \boldsymbol{I}$, then according to the trace properties, we have:

$$\begin{aligned} \operatorname{tr}(\boldsymbol{W}_2^\top \boldsymbol{Z}^\top \boldsymbol{Z}\boldsymbol{W}_2) &= \operatorname{tr}(\boldsymbol{W}_2 \boldsymbol{W}_2^\top \boldsymbol{Z}^\top \boldsymbol{Z}) \\ &= \operatorname{tr}(\boldsymbol{Z}^\top \boldsymbol{Z}). \end{aligned} \tag{19}$$

Then we have:

$$\begin{aligned} \hat{\boldsymbol{W}}_{\boldsymbol{2}} &= \arg\min_{\boldsymbol{W_2}} \operatorname{tr}(\boldsymbol{T}^\top \boldsymbol{T}) + \operatorname{tr}(\boldsymbol{W}_2^\top \boldsymbol{Z}^\top \boldsymbol{Z}\boldsymbol{W}_2) - 2\operatorname{tr}(\boldsymbol{T}^\top \boldsymbol{Z}\boldsymbol{W}_2)) \\ &= \arg\min_{\boldsymbol{W_2}} \operatorname{tr}(\boldsymbol{T}^\top \boldsymbol{T}) + \operatorname{tr}(\boldsymbol{Z}^\top \boldsymbol{Z}) - 2\operatorname{tr}(\boldsymbol{T}^\top \boldsymbol{Z}\boldsymbol{W}_2)). \end{aligned} \tag{20}$$

Since we fix $\boldsymbol{T}$ and $\boldsymbol{S}$ to obtain a closed solution $\boldsymbol{W}_2$, we only need to consider $\operatorname{tr}(\boldsymbol{T}^\top \boldsymbol{Z}\boldsymbol{W}_2))$.

$$\begin{aligned} \hat{\boldsymbol{W}}_{\boldsymbol{2}} &= \arg\min_{\boldsymbol{W_2}} -\operatorname{tr}(\boldsymbol{T}^\top \boldsymbol{Z}\boldsymbol{W}_2)) \\ &= \arg\max_{\boldsymbol{W_2}} \operatorname{tr}(\boldsymbol{T}^\top \boldsymbol{Z}\boldsymbol{W}_2)), \end{aligned} \tag{21}$$

we conduct svd decompostion on $\boldsymbol{T}^\top \boldsymbol{Z} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$, then we have:

$$\begin{aligned} \hat{\boldsymbol{W}}_{\boldsymbol{2}} &= \arg\max_{\boldsymbol{W_2}} \operatorname{tr}(\boldsymbol{T}^\top \boldsymbol{Z}\boldsymbol{W}_2)) \\ &= \arg\max_{\boldsymbol{W_2}} \operatorname{tr}(\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top \boldsymbol{W}_2) \\ &= \arg\max_{\boldsymbol{W_2}} \operatorname{tr}(\boldsymbol{\Sigma}\boldsymbol{V}^\top \boldsymbol{W}_2\boldsymbol{U}). \end{aligned} \tag{22}$$

Since $\boldsymbol{\Sigma}$ is a diagonal matrix where the diagonal are all singular values. Then the best solution is making $\boldsymbol{V}^\top \boldsymbol{W}_2 \boldsymbol{U} = \boldsymbol{I}$ as well. Then we have the solution, $\boldsymbol{W}_2 = \boldsymbol{V}\boldsymbol{U}^\top$. And our distillation objective is to maximize $\operatorname{tr}(\boldsymbol{\Sigma})$. □

Implementation details. We introduce details of KD methods' training settings here. Based on the basic settings introduced in the experiment, we mainly follow the official implementation of (Tian et al., 2020) for the compared KD methods. We set batch size 512 with a learning rate of 4×10^{-3} for VGG8 and batch size $1024$with a learning rate of 8×10^{-3} for other student models to utilize the GPU memory fully. Each model will extract a 5-stage feature for mapping and distillation. For the

Table 9: Self-implementation results of CRD and MGD

Teacher Student	Swin MobileNetV2	Swin MobileNetV3	Swin VGG8	Swin ResNet18	Swin ShuffleNetV1	Swin ShuffleNetV2	Avg. Avg.
Teacher	94.48%	94.48%	94.48%	94.48%	94.48%	94.48%	94.48%
Student	84.70%	86.44%	78.86%	85.24%	77.30%	79.52%	82.01%
CRD	82.32%	70.56%	77.24%	84.16%	74.50%	77.02%	77.63%
CRD†	85.44%	86.18%	79.16%	85.78%	74.72%	77.88%	81.52%
MGD	84.52%	85.48%	78.72%	85.62%	77.98%	79.62%	81.99%
MGD†	84.48%	86.36%	78.38%	85.42%	77.68%	79.46%	81.96%

Table 10: Top-1 and Top-5 accuracies of student network MobileNetV2 on ImageNet validation set. We use SwinTransformer-tiny, released by the official repository, as the teacher network. For all KD methods, we adopt the same training protocol.

	Teacher	Student	AT	SP	RKD	FitNet	CRD	KD	Ours
Top-1	80.90%	71.88%	71.73%	71.97%	71.00%	72.07%	71.93%	72.04%	**72.20%**
Top-5	96.04%	90.36%	90.49%	90.50%	89.92%	90.60%	90.72%	90.49%	**90.56%**

FitNet (Romero et al., 2014) implementation, kernel sizes are modified to align the downsample rate and reduce computation complexity. In Review (Chen et al., 2021), we set the middle dimension of all experiments to 256. In CRD (Tian et al., 2020), we notice that the original implementation is ineffective in the vision transformer distillation scenario. We only use one encoder projection where the student representation is mapped to teacher representation of the same size, i.e., 1024. We also adopt the InfoNCE loss introduced in (Oord et al., 2018; He et al., 2019). The original results can be found in Table 9. We also make some ablation studies on MGD (Yang et al., 2022) where we adopt masked generative module on representation level. As shown in Table 9, both intermediate feature-level and representation-level MGD do not obtain good results. Moreover, both of them perform similarly. † means our self-implementation. We slightly change the training setting in ImageNet1k. The optimizer AdamW is replaced with SGD, and the learning rate is set to 0.4 with an effective batch size 1024. We also drop some strong augmentations such as auto-augmentations. Even so, our baseline is still much stronger than the results in the original paper. All experiments are conducted on 4 NVIDIA 3090 GPUs.

As shown in Table 6, we also compare our our ImageNet1K result on MobileNetV2. Our results still achieve the best performance.

Visualization. We also use CKA introduced in (Kornblith et al., 2019; Raghu et al., 2021) to visualize the representation similarity of different architectures. As shown in Figure 4, we can find that when we use ImageNet1K pretrained models provided by PyTorch. The ResNet18 performs very similarly to ResNet50. However, the resent50 did not look similar to Vit-Base. However, after distillation, the two different structures seem more positively correlated. The formulation of CKA is:

$$\mathrm{CKA}\left(\boldsymbol{K}, \boldsymbol{L}\right)=\frac{\mathrm{HSIC}\left(\boldsymbol{K}, \boldsymbol{L}\right)}{\sqrt{\mathrm{HSIC}\left(\boldsymbol{K}, \boldsymbol{K}\right) \mathrm{HSIC}\left(\boldsymbol{L}, \boldsymbol{L}\right)}}, \tag{23}$$

where HSIC is defined as:

$$\mathrm{HSIC}\,\boldsymbol{K}, \boldsymbol{L})=\frac{1}{(n-1)^2}\,\mathrm{tr}\left(\boldsymbol{K H L H}\right). \tag{24}$$

$\boldsymbol{H}$ is the centering matrix, $\boldsymbol{K}$ and $\boldsymbol{L}$ are student and teacher output, respectively. CKA is a tool to measure representation similarities and has been widely used in measuring the similarities between Vision transformers and CNNs Raghu et al. (2021). As shown in 4, we can find that when we use ImageNet1K pretrained models provided by PyTorch. The ResNet18 performs very similarly to ResNet50. However, the resent50 did not look similar to Vit-Base.

Discussion. More discussions are provided here. Firstly, we would like to discuss the effectiveness of current CNN methods in Vision transformers distillation scenarios. Except for architecture gaps, another possible reason that CNNs, especially tiny CNNs, are poor at capturing global features

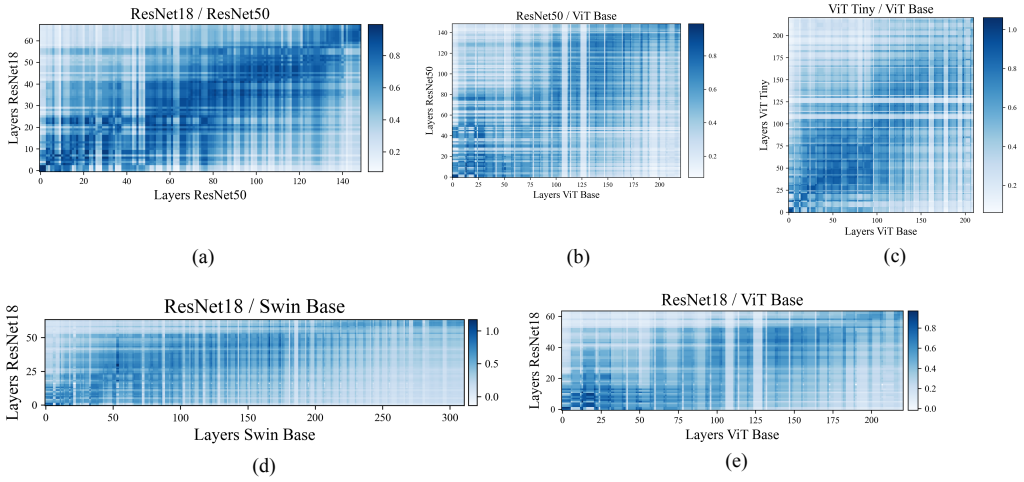


Figure 4: Representation similarities comparison between different architecture. (a) The representation similarity between ResNet18 and ResNet50 with ImageNet1K pretrained model. (b) The representation similarity between ResNet50 and ViT-base with ImageNet1K pretrained model. (c) The representation similarity between ViT-tiny and ViT-base with ImageNet1K pretrained model. (d) The representation similarity between ResNet50 and SwinTransformer-base after distillation. (e) The representation similarity between ResNet18 and ViT-base after distillation.

as it requires a large effective receptive field (Luo et al., 2016; Ding et al., 2022). Thus distilling global information somehow crushes tiny CNNs' training pattern because the path of capturing local features is redundant and noisy. In contrast, the path of capturing global features is low-frequency. A general idea to solve the problem is to modify the backbone of tiny CNNs, such as using large kernel sizes, different normalizations, and fewer activations as introduced in ConvNext (Liu et al., 2022). However, we tend to keep the original tiny CNNs' architecture in the distillation scenario. Then we propose an encoder-decoder representation distillation structure to address the issue.

We would also like to compare the performance differences between Vision Transformer distillation and CNN distillation. As shown in Table 11 and Table 12, we can find that with different architectures, previous KD methods mainly show their effectiveness in distilling CNNs to CNNs. Note that we directly use the results from CRD (Tian et al., 2020). This comparison indicates that the tiny CNNs are more difficult to learn global information than to capture other local information.

Table 11: Top-1 *accuracy* of teacher and student networks on ImageNet100 of 11 KD methods with a Vision Transformer teacher. † means self-implementation.

Teacher Student	Swin MobileNetV2	Swin MobileNetV3	Swin VGG8	Swin ResNet18	Swin ShuffleNetV1	Swin ShuffleNetV2	Avg. Acc.
Teacher	94.48%	94.48%	94.48%	94.48%	94.48%	94.48%	94.48%
Student	84.70%	86.44%	78.86%	85.24%	77.30%	79.52%	82.01%
KD	85.34%	86.82%	79.04%	85.54%	77.46%	79.86%	82.34%
DKD	84.90%	86.86%	78.70%	85.96%	78.30%	80.02%	82.45%
AT	82.92%	66.86%	76.90%	84.80%	74.92%	76.74%	77.19%
SP	83.82%	85.26%	74.30%	84.66%	74.16%	76.12%	79.71%
RKD	78.68%	85.06%	76.52%	85.24%	75.42%	77.48%	79.73%
PKT	84.32%	86.84%	76.82%	85.20%	76.96%	78.86%	81.50%
FitNet	84.70%	86.44%	78.86%	85.24%	77.30%	79.52%	82.01%
VID	85.02%	86.46%	78.94%	86.10%	77.92%	80.06%	82.42%
CRD	82.32%	70.56%	77.24%	84.16%	74.50%	77.02%	77.63%
CRD†	85.44%	86.18%	79.16%	85.78%	74.72%	77.88%	81.52%
Review	84.36%	86.96%	78.90%	86.74%	78.62%	79.80%	82.56%
MGD	84.52%	85.48%	78.72%	85.62%	77.98%	79.62%	81.99%

Table 12: Top-1 test *accuracy* of student networks on CIFAR100 of a number of distillation methods with various CNN teachers.

Teacher Student	VGG13 MobileNetV2	ResNet50 MobileNetV2	ResNet50 VGG8	ResNet32x4 ShuffleNetV1	ResNet32x4 ShuffleNetV2	WRN-40-2 ShuffleNetV1	Avg. Acc.
Teacher	74.64%	79.34%	79.34%	79.42%	79.42%	75.61%	77.96%
Student	64.6%	64.6%	70.36%	70.5 %	71.82%	70.5%	68.73%
KD	67.37%	67.35%	73.81%	74.07%	74.45%	74.83%	71.98%
FitNet	64.14%	63.16%	70.69%	73.59%	73.54%	73.73%	69.81%
AT	59.40%	58.58%	71.84%	71.73%	72.73%	73.32%	67.93%
SP	66.30%	68.08%	73.34%	73.48%	74.56%	74.52%	71.71%
CC	64.86%	65.43%	70.25%	71.14%	71.29%	71.38%	69.06%
VID	65.56%	67.57%	70.30%	73.38%	73.40%	73.61%	70.64%
RKD	64.52%	64.43%	71.50%	72.28%	73.21%	72.21%	69.69%
PKT	67.13%	66.52%	73.01%	74.10%	74.69%	73.89%	71.56%
AB	66.06%	67.20%	70.65%	73.55%	74.31%	73.34%	70.85%
FT	61.78%	60.99%	70.29%	71.75%	72.50%	72.03%	68.22%
NST	58.16%	64.96%	71.28%	74.12%	74.68%	74.89%	69.68%
CRD	69.73%	69.11%	74.30%	75.11%	75.65%	76.05%	73.33%