Knowledge Distillation Layer that Lets the Student Decide

Ada Görgün*
ada.gorgun@metu.edu.tr
Yeti Z. Gürbüz*
yeti@metu.edu.tr
A. Aydın Alatan
alatan@metu.edu.tr

Dept. of Electrical and Electronics Eng.
& Center for Image Analysis (OGAM)
Middle East Technical University
Ankara, Turkey

Abstract

Typical technique in knowledge distillation (KD) is regularizing the learning of a limited capacity model (*student*) by pushing its responses to match a powerful model's (*teacher*). Albeit useful especially in the penultimate layer and beyond, its action on student's feature transform is rather implicit, limiting its practice in the intermediate layers. To explicitly embed the teacher's knowledge in feature transform, we propose a learnable KD layer for the student which improves KD with two distinct abilities: *i)* learning how to leverage the teacher's knowledge, enabling to discard nuisance information, and *ii)* feeding forward the transferred knowledge deeper. Thus, the student enjoys the teacher's knowledge during the inference besides training. Formally, we repurpose 1x1-BN-ReLU-1x1 convolution block to assign a semantic vector to each local region according to the template (supervised by the teacher) that the corresponding region of the student matches. To facilitate template learning in the intermediate layers, we propose a novel form of supervision based on the teacher's decisions. Through rigorous experimentation, we demonstrate the effectiveness of our approach on 3 popular classification benchmarks. Code is available at: letKD Framework

1 Introduction

The unprecedented success of convolutional neural networks (CNN) having massive computational and memory complexity has shaped the efforts to find a compromise between the model size and the performance for the effective deployment in devices with limited resources. Knowledge distillation (KD) [2, 16] is a complementary method to those efforts including lightweight model design [5], and model compression techniques such as model pruning [22, 23] or quantization [36]. KD is built on boosting the performance of a relatively smaller model (*i.e.*, *student*) by leveraging the knowledge encoded in a powerful model (*i.e.*, *teacher*).

The two critical questions in KD are *how* and *in which form* to transfer the knowledge so that it can benefit the student the most [8]. Although, the prolific

*Equal contribution.

and varied literature of KD includes diverse forms of knowledge to transfer through regressing the predictions [16, 25], the intermediate representations [14, 29, 40], and the metrics induced by the distances among the sample representations of either the penultimate layer [26, 33] or the intermediate layers [34]; all of these methods have a shared component to facilitate student's learning: regularization with a discrepancy loss between the matching targets.

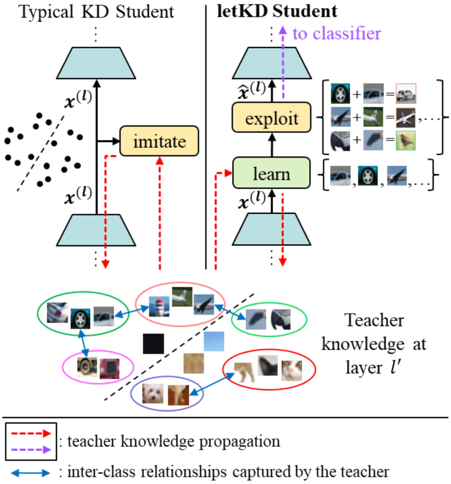

Figure 1: The differences between training the student with typical KD methods and with letKD.

Common intuition to explain the effectiveness of such regularization is by considering the soft targets learned from a teacher to capture the missing relationships among different categories that sole label supervision cannot provide [16]. This intuition can apply well while employing the regularization to the predictions at the penultimate layer since both the student and the teacher share the model beyond, *i.e.*, a linear classifier. However, its solidity for the intermediate layers is questionable. Isn't it demanding to force the student to imitate the teacher due to their architectural differences? Surely, there exist methods that propose feature alignment modules [4, 29] to match the feature dimensions between the teacher and the student. Still, the student can fail to exploit those intermediate representations as effectively as the teacher having more layers on top does. Indeed, empirical studies show that knowledge transfer is more effective in the penultimate layer than intermediate layers [18]. Although this issue is partially addressed by changing the form of the knowledge into the teacher's coarse decisions which the student can comprehend [31], its regularization of student's learning is still not that effective. Then, is sole regularization the best choice to transfer knowledge? Moreover, can we explicitly use the teacher's knowledge in the inference as well? In this paper, we try to address those questions within the context of a feature extraction process. We propose a learnable feature transform layer that effectively lets the student decide whether to leverage the teacher's knowledge and use it explicitly during the inference in addition to regularizing the learning.

Specifically, inspired by [7], we revamp 1x1-BN-ReLU-1x1 based feature transform to assign a feature vector to each local region according to the semantic meaning of the template that the corresponding region matches. We propose to supervise the templates with the semantic entities learned by the teacher and leave the semantic vector learning to the student. This will let the student learn the entities (*e.g.*, *wing*, *tire*) that the teacher finds useful and exploit them (*e.g.*, *wing* and *beak* $\rightarrow$ *bird*) in feature transform, enabling us to feed forward the knowledge rather than imitating it as in Fig. 1. To enable learning, we employ a soft-max solution to the best-matching template and represent its feature vector as the weighted combination of the semantic vectors with the matching scores. We rigorously let the solution space include 0-weight, effectively enabling the student to discard transferred knowledge. Hence, the student is not only able to reshape the transferred knowledge with its semantic vectors but also feed forward it to the upper layers, which is a novel approach in KD.

To validate our claims, we design an extensive empirical study. The results confirm that feed forwarding the teacher's knowledge by explicitly using it in feature transform improves student models, and our layer enables the utilization of the teacher's knowledge during inference. We tested our method on 10 student models and 3 classification benchmarks, showing its wide applicability. We set a new state-of-the-art by consistently improving upon the direct application of multi-layer teacher supervision [18, 31] and other KD methods in both single and multi-layer transfer settings.

2 Related Work

Our contributions. Prior to discussing the works that are most related to ours, we recapitulate our contributions as *i*) we propose a learnable KD layer that captures the teacher's knowledge during training and employs it in feature transform, effectively feeding forward the transferred knowledge deeper and enjoying it during the inference as well, *ii*) we repurpose some convolution kernels of the teacher as the cluster centers to semantic entities and exploit them in KD, *iii*) we introduce a novel form of supervision based on the teacher's decisions on the intermediate layers.

KD in penultimate layer. Leading momentum in KD is built on transferring the inter-category relations captured by the teacher. Thus, many works regularize the student's learning by matching its predictions with the teacher's soft predictions [16, 25, 35, 43]. Following a similar perspective, relations among the local features are also exploited as the teacher's knowledge, which include the metrics induced by the distances among the sample representations [26, 33, 38] and fine-grained labeling obtained by clustering the teacher's features [18]. KD regularization in the penultimate layer typically outperforms its intermediate layer counterparts since both the teacher and the student have the same representation power on top, *i.e.*, a linear classifier. Indeed, recent work [3] shows that the student can directly copy the teacher's classifier once their features match. In our work, we also build on teacher's soft labeling for KD. Differently, we do not employ it as a regularizer. We instead store them within the parameters of our KD layer and exploit them in feature transform.

KD in intermediate layers. Lacking category-based annotations from the teacher, KD at lower layers uses other forms of supervision including local features [15, 29], saliency maps [40], feature distributions [1, 4], and the metrics induced by the inter-feature distances [28, 34]. Such methods push the student to imitate the geometry of the teacher's intermediate representations. However, the intuition of capturing such relations does not directly apply in the intermediate layers as it does in the penultimate layer. The student can fail to exploit those intermediate representations as effectively as the teacher having more layers on top does. Although recent works explicitly study mitigating this problem by supervising penultimate layer using multiple intermediate layers with improved feature alignment modules [4], selectively deciding which intermediate layers to distill [45] or changing the form of the knowledge into the teacher's coarse decisions [31], it is still empirically observed that including KD in intermediate layers has a negative effect [18]. In contrast to existing efforts, we profitably exploit the teacher's knowledge in the intermediate layer with our KD layer. Our approach enables the student to build new representations from the semantic entities learned by the teacher as well as to discard the nuisance information, which differs from existing selective feature distillation schemes [37].

Deeply supervised nets. Our student's learning scheme is closely related to the methods that use auxiliary classification loss to regularize the features and to facilitate learning without vanishing gradients [21, 32]. Differently, we explicitly use such intermediate predictions to semantically represent local regions with the combination of learned vectors weighted by those predictions. Similar to us, class-level predictions are used to shape the behavior of the intermediate features in [7] for the classification problem. Different from them, we relate such a mechanism to KD for the first time and we additionally propose a novel form of supervision that is based on the local decisions of the teacher, yielding effective sub-class annotations.

3 Preliminaries

Consider the mapping $f(\cdot;\theta): \mathcal{X} \to \mathcal{Y}$ within L-layer composite function family, *i.e.*, $f = f_L \circ f_{L\text{-}1} \circ \cdots f_2 \circ f_1$, where $\mathcal{X}$ is the space of data points, $\mathcal{Y}$ is the space of labels, and θ is the model parameters. We consider two models, *i.e.*, the student $f_s(x;\theta_s)$, and the teacher $f_t(x;\theta_t)$, with $|\theta_t| > |\theta_s|$. Given samples $\{x_i\}_i \sim \mathcal{X}$, we let $\mathcal{S}_l = \{f_s^{(l)}(x_i;\theta_s)\}_i$ and $\mathcal{T}_{l'} = \{f_t^{(l')}(x_i;\theta_t)\}_i$ denote the set of student features and the teacher features at layer l and l', respectively with $f^{(l)} = f_l \circ f_{l\text{-}1} \circ \cdots \circ f_2 \circ f_1$. We consider the following matching cost:

$$\mathcal{L}_{KD}(\mathcal{S}_l, \mathcal{T}_{l'}) = \|g_s(\mathcal{S}_l) - g_t(\mathcal{T}_{l'})\|_{\mathcal{M}} \tag{3.1}$$

where $\| \cdot_1 - \cdot_2 \|_{\mathcal{M}}$ is a metric to compare its arguments, g_s and g_t are the transformations to match the dimensions of $\mathcal{S}_l$ and $\mathcal{T}_{l'}$. For example, when $\mathcal{S}_l$ and $\mathcal{T}_{l'}$ correspond to model predictions [16], g_s and g_t are *scaled soft-max*, and $\mathcal{M}$ is *KL-div*.

Given the samples $\{(x_i, y_i)\}_i \sim \mathcal{X} \text{x} \mathcal{Y}$ for the classification task, typical KD methods regularize the student's learning with $\mathcal{L}_{KD}$ to transfer the teacher's knowledge as:

$$\mathcal{L}(\theta_s) = \mathcal{L}_{CE}(\{(f_s(x_i;\theta_s), y_i)\}_i) + \lambda \mathcal{L}_{KD}(\mathcal{S}_l, \mathcal{T}_{l'}) \tag{3.2}$$

where $\mathcal{L}_{CE}$ is the *cross-entropy* loss and λ is the weight of the distillation loss. In the rest of the paper, we additionally consider employing the teacher's knowledge in feature transform as $f = \cdots f_{l+1} \circ f_{KD} \circ f_l \cdots$ with f_{KD} as a function of $\mathcal{T}_{l'}$ and propose a learnable layer to use it without teacher's feedback $\mathcal{T}_{l'}$ in the inference.

4 Method

We propose a lightweight residual layer with 1x1-BN-ReLU-1x1 convolution block for KD. Our layer enhances its input feature map with the knowledge transferred from the teacher. We first explain our theoretical motivation in § 4.1. The theory suggests that we can use the teacher's knowledge rather explicitly in feature extraction and feed forward it deeper if the kernels of the first 1x1 convolutions are guided by the teacher's supervision. We use our layer in both the penultimate layer and an intermediate layer. To facilitate learning, we propose different teacher supervision mechanisms for both in § 4.2, where we propose a novel form of supervision that is based on the teacher's decisions.

4.1 Learnable KD Layer

Our KD technique is built on the perspective [9, 10, 41, 44, and references therein], explaining the success of the CNNs by considering each pixel of a feature map as a semantic entity (*e.g.*, *wing*, *tire*, *etc.*). We aim to transfer such semantic entities learned by the teacher to the student. Let $x^{(l)}$ denote a hxw feature map after the l^{th} layer and $x_i^{(l)} \in \mathbb{R}^d$ is its i^{th} spatial feature. Our KD layer depicted in Fig. 2 transforms the feature at each pixel-i as:

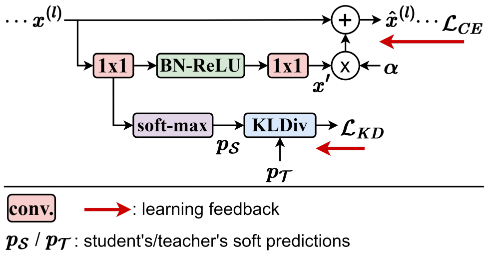

Figure 2: Proposed KD layer.

$$\hat{x}_i^{(l)} = x_i^{(l)} + \alpha x_i' \tag{4.1}$$

where $x_i' = g(x^{(l)})_i$ is the i^{th} spatial feature of the map after we apply g, and $\alpha > 0$ is a scalar constant. We implement g as 1x1-BN-ReLU-1x1 where 1x1 is convolution with unit spatial extent, *i.e.*, linear transform. We leverage the teacher's knowledge to supervise the learning of the first convolution kernels. We now technically explain how such a simple residual block is an effective way for knowledge transfer.

Motivation. We first repurpose $g \coloneqq$ 1x1-BN-ReLU-1x1 transform as feature embedding by template matching, following [7]. Specifically, we first show that g selects the best-matching kernel to $x_i^{(l)}$ and assigns a feature vector to pixel-i according to the semantic meaning of the matched kernel. We then regularize the learning of the matching kernels by predicting the fine-grained labels (*e.g.*, *wing*, *tire*, *etc.*) provided by the teacher. Namely, the matching kernels become the weights of a linear classifier (*i.e.*, class representatives). In the end, the predictions match the teacher's decisions for local regions and the student assigns semantic vectors based on those predictions. Hence, the student controls how to use the transferred knowledge. Moreover, the knowledge is explicitly embedded in the feature transform, effectively enabling its use during inference.

Formulation. For CNNs, $x_i^{(l)}$ represents a local region around it to some spatial extent depending on the depth. To simplify notation, we use $x_i = x_i^{(l)}$ henceforth. We consider a set of matching kernels $\{\omega_k \in \mathbb{R}^d\}_k$ as templates, each of which seeks for a particular pattern. To each kernel ω_k, we associate an embedding vector $\nu_k \in \mathbb{R}^{d'}$ representing the semantics of the corresponding pattern. We aim to replace x_i with the embedding vector of its best-matching kernel. We formally write this process as:

$$p_{|i} = \arg\max_{p,q \geqslant 0} q\,\mu + \Sigma_k p_k\, \omega_k^\intercal x_i \quad \text{s.to} \quad q + \Sigma_k p_k = 1 \tag{4.2}$$

where μ is a threshold enabling to zero out the embedding vector if no kernel matches with at least μ similarity. Then, we assign the representation of x_i as $x_i' = \Sigma_k p_{k|i}\, \nu_k$ since $p_{|i}$ is either one-hot or zero vector owing to *total unimodularity* [17]. To enjoy analytical gradients, we employ entropy smoothing to the objective in (4.2) as:

$$p_{|i} = \arg\max_{p,q \geqslant 0} q\,\mu + p^\intercal a_{|i} - \tfrac{1}{\epsilon}(q \log q + p^\intercal \log p) \quad \text{s.to} \quad q + \Sigma_k p_k = 1 \tag{4.3}$$

and obtain a soft-max solution $p_{k|i} = \frac{\exp(\epsilon a_{k|i})}{\exp(\epsilon\mu)+\Sigma_{k'}\exp(\epsilon a_{k'|i})}$ where $a_{k|i} = \omega_k^{\intercal} x_i$ and ϵ controls the smoothness of $p_{|i}$. Hence, we can implement the feature embedding by template matching via 1x1-SoftMax-1x1 with $\{\omega_k\}_k$ and $\{\nu_k\}_k$ as the convolution kernels. Finally, when we obtain this smooth labeling from the teacher as $p_{\mathcal{T}}(i)$ for each pixel-i, we can regularize the learning of $\{\omega_k\}_k$ by minimizing the *KL-div* between the predictions of the student $p_{\mathcal{S}}(i) \coloneqq \mathrm{softmax}(a_{|i})$ and $p_{\mathcal{T}}(i)$ for all pixels as $\mathcal{L}_{KD}$ in (3.1) with $\lambda = 1$.

Practical simplifications. With the smooth assignments obtained by (4.3), the student shapes the teacher's knowledge by the weighted combination of its embedding vectors $\{\nu_k\}_k$ with the weights proportional to the matching scores. To enable the student to discard nuisance information from the teacher, we must set a proper μ. As empirically validated in [7] as well as formally discussed in the supplementary material [11, Appendix], we can indeed inherently learn it from batch statistics by replacing soft-max solution of $p_{k|i}$ in (4.3) with BN-ReLU, *i.e.*, $p_{|i} \approx \text{BN-ReLU}(a_{|i})$.

4.2 Teacher Supervision

A critical desiderata of our KD layer is per pixel label annotations, $p_{\mathcal{T}}(i)$, provided by the teacher. We propose different forms of teacher supervision for the uses in the penultimate layer and the intermediate layers. In the following, we assume that the spatial dimensions of the teacher's and the student's feature maps match at the layers where the knowledge transfer is performed, noting that we can transform the student's feature map with *pooling, strided convolution, etc.* to match the dimensions.

4.2.1 Penultimate Layer

Expanding on the idea [11] that we view each feature map pixel as a semantic entity, we propose to employ K-means to the teacher's features at the penultimate layer to obtain fine-grained labels for the semantic entities. We then annotate each pixel by soft-max assignments to the cluster centers to capture inter-category relations.

Formally, given the dataset samples $\{x_i\}_i \sim \mathcal{X}$ (or a subset of it), we compute the teacher's feature maps at the penultimate layer, *i.e.*, $\{f_t^{(-1)}(x_i;\theta_t) \in \mathbb{R}^{h \times w \times d}\}_i$. Considering each pixel as a feature sample, we fit K-means clustering to the pixels of those maps and obtain the centers $\{\rho_k\}_{k\in[K]}$ where $[K] = 1,\dots,K$. During training, we pass the input, x, through the teacher to obtain $f_t^{(-1)}(x;\theta_t)$. We then compute the distance $d_{k|i} = \|f_t^{(-1)}(x;\theta_t)_i - \rho_k\|_2^2$ for each pixel-i of the feature map to the cluster centers and obtain their K-dimensional soft labeling as $p_{\mathcal{T}}(i) = \mathrm{softmax}(d_{|i})$. Although this form of supervision is previously applied to KD [18] to match the responses as in (3.1), we differently repurpose it as a supervision for our matching kernels and enable the student to rather exploit it in feature transform than imitate through our KD layer (Fig. 2). We summarize our approach in the supplementary material [11, § 3] with Algorithms 1 and 3.

Without K-means. We can directly obtain the soft assignments to the semantic entities from the architectures (*e.g.*, ResNet [12, 13]) that involve similar blocks to 3x3-BN-ReLU-1x1 by design. Building on our analysis in § 4.1, kernels of 3x3 correspond to learnable templates (*i.e.*, cluster centers) of some semantic entities. Hence, we can use the soft-maxed activations of the 3x3 convolution at the final block to obtain $p_{\mathcal{T}}$. Supporting our claims in § 4.1, we empirically show in Tab. 3 that this clustering-free approach has competitive performance with much fewer centers compared to K-means.

4.2.2 Intermediate Layer

Granted that we can apply K-means based supervision in § 4.2.1 to intermediate layers as well, its data-driven annotation mechanism corresponds to representing the geometry of the teacher's features at some point. Although this is beneficial in the penultimate layer due to the shared classifier architecture afterward, it cannot be effectively used by the student's intermediate layers due to capacity differences. In fact, we empirically observe in Tab. 4 that such supervision without our KD layer is detrimental to the performance. Although our KD layer is a remedy thanks to not pushing the student to imitate but to exploit the knowledge, it can benefit more from another form of supervision. In particular, decision-based supervision [31] has recently been shown to be superior to representation-based, yet it is tailored for coarse decisions of the teacher at the intermediate layers. Thus, as summarized in the supplementary material [1, § 3] with Algorithm 2 as well as visualized in Fig. 3, we now propose a new supervision based on localized fine-grained decisions to facilitate the learning of our KD layer.

To transfer teacher's localized decisions, we enhance K-means based annotations by exploiting the original labels of the image. Specifically, we compute the teacher maps at layer-l', *i.e.*, $\{f_t^{(l')}(x_i;\theta_t) \in \mathbb{R}^{h \times w \times d}\}_i$, for the samples $\{(x_i, y_i)\}_i$ in the dataset (or a subset of it). Considering each pixel of $f_t^{(l')}(x_i;\theta_t)$ as a feature sample with class label y_i, we fit a linear classifier using linear discriminant analysis (LDA) to set the stage for the rest of the formulation. Once fitted, LDA is simply a 1x1 convolution (*i.e.*, per pixel linear transform) that aims to map the features close only if they share the same label, reflecting localized decision capacity of the teacher at layer-l'.

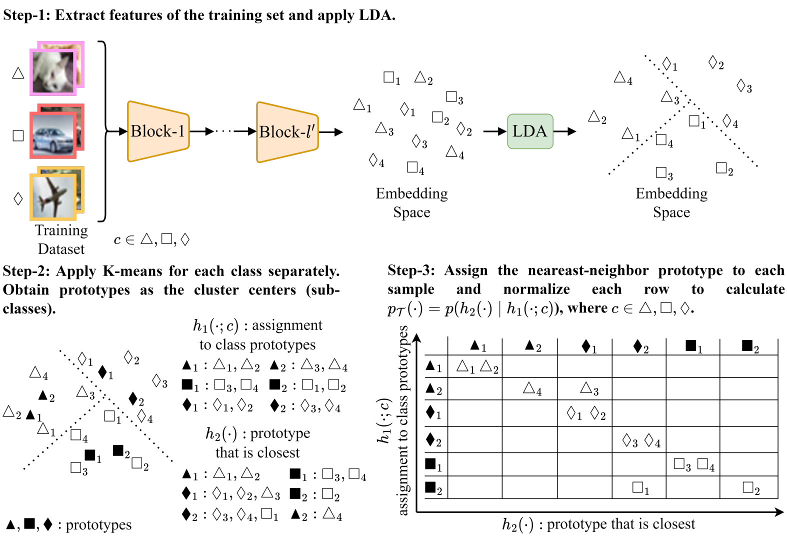

Figure 3: Visualization of the teacher's intermediate layer supervision.

Surely, we expect some features to be shared among different classes (*e.g.*, *tire* for *truck* and *car*) while some are discriminative (*e.g.*, *beak* for *bird*). To capture such localized decisions, we apply K-means of small K to the features of each class separately and obtain K-many sub-class centers for each, *i.e.*, we have $K{\times}C$-many centers for a C-class problem. Next, we consider two nearest-neighbor classifiers: $h_1(\cdot;c)$ that labels its input with the ID of the nearest center belonging to class $c{\in}[C]$, and $h_2(\cdot)$ that assigns the ID of the nearest center to its input. Using the dataset, we estimate the label distribution of the features belonging to a particular sub-class. Namely, we estimate the probability that h_2 assigns j when h_1 assigns i and obtain the distribution $p(h_2(\cdot)\mid h_1(\cdot;c))$, which allows us to see whether the teacher finds it useful to discriminate some sub-classes at layer-l'. During training, we pass the input, (x,y), through the teacher to have $z \coloneqq \mathrm{LDA}(f_t^{(l')}(x;\theta_t))$, and obtain the soft labeling for each pixel-i of the feature map as $p_{\mathcal{T}}(i) = p(h_2(z_i)\mid h_1(z_i,y))$ (*i.e.*, *rows* of the table in Fig. 3). Different from data-driven annotations, we use the same annotation for all the features belonging to the same sub-class. Hence, the transferred knowledge represents the teacher's local decision capacity rather than its feature geometry.

5 Experimental Work

Reducing the confounding of the factors other than our method, we adopted the framework implemented by [18] in PyTorch [27] to make a fair and unbiased evaluation of our method as well as comparisons with the other invented methods. We evaluated our method on CIFAR-100 [19], Tiny-ImageNet [20] and ImageNet [6] with various architectures including ResNet (RN) [12, 13], Wide ResNet (WRN) [39], MobileNet (MNV2) [30] and ShuffleNet (SNV1/V2) [24, 42]. We attribute our methods as letKD-1 and letKD-2, where letKD-1 represents the single penultimate layer KD while letKD-2 denotes the inclusion of intermediate layer KD. As for selecting the location of the layers in our methods, we deliberately used the penultimate layer, a common choice in KD methods, as it carries the most discriminative and higher-level information. For the intermediate layer, we relied on TDD's findings [31] indicating class-agnostic characteristics of lower layer teacher's features and thus, omitted the lower layers and decided to place our KD layer after the first residual block of the last stage for all networks, and used the output of the first block at the last stage of the teacher. Additionally, this choice ensures enough receptive field for the validity of our template matching formulation defined in (4.2). We defer further empirical details including hyperparameter selection and its analysis to the supplementary material [11].

5.1 Results

We provide the results in Tabs. 1 and 2 for the evaluations on CIFAR-100, and ImageNet, respectively. We compare our method against KD [16], FitNet [29], DKD [43], SimKD [3], TDD [31] and QUEST [18]. We defer the results for Tiny-ImageNet and the extended versions of Tabs. 1 and 2 with other KD alternatives to the supplementary material [11, § 1.1].

CIFAR-100 (Tab. 1). Our method letKD-2 outperforms all the other methods on all different teacher-student combinations with homogeneous and heterogeneous architecture settings with letKD-1 being the second best, except for RN32x4-RN8x4 where we are slightly behind SimKD [3]. With that being said, SimKD includes an additional 1x1-3x3-1x1 convolution block before the penultimate layer, which captures further local relations. When we compare our method with their 1x1-1x1 version, we outperform them by $\approx 1.3\%$ points margin. Moreover, SimKD is inferior in all the other teacher-student pairs even with their largest block, yet another supporting evidence for the effectiveness of our KD layer.

Table 1: Top-1 acc. averaged over 5 trials on CIFAR100. **Bold**: best in its category.

Archs. →	**Homogeneous**						**Heterogeneous**			
Teacher Student	WRN-40-2 WRN-16-2	WRN-40-2 WRN-40-1	RN56 RN20	RN110 RN20	RN110 RN32	RN32x4 RN8x4	WRN-40-2 SNV1	RN32x4 SNV1	RN32x4 SNV2	RN50 MNV2
Methods ↓	75.61 73.26	75.61 71.98	72.34 69.06	74.31 69.06	74.31 71.14	79.42 72.50	75.61 70.50	79.42 70.50	79.42 71.82	79.34 64.60
KD	74.92	73.54	70.66	70.67	73.08	73.33	74.83	74.07	74.45	67.35
FitNet	73.58	72.24	69.21	68.99	71.06	73.50	73.73	73.59	73.54	63.16
DKD	76.24	74.81	71.97	-	74.11	76.32	76.70	76.45	77.07	70.35
SimKD	76.06	74.92	68.95	69.35	72.15	**78.08**	76.95	77.18	77.78	68.91
TDD	75.01	74.04	71.53	-	-	-	75.60	-	-	68.37
QUEST	76.10	74.58	71.84	71.89	74.08	75.88	76.75	76.28	77.09	69.81
letKD-1	76.29 $\mp$0.15	75.01 $\mp$0.09	72.44 $\mp$0.24	72.68 $\mp$0.31	74.40 $\mp$0.14	76.70 $\mp$0.06	76.93 $\mp$0.16	76.65 $\mp$0.24	77.75 $\mp$0.17	69.97 $\mp$0.18
letKD-2	**76.56 $\mp$0.22**	**75.19 $\mp$0.13**	**73.27 $\mp$0.16**	**73.38 $\mp$0.14**	**74.62 $\mp$0.20**	**77.09 $\mp$0.18**	**77.08 $\mp$0.12**	**77.30 $\mp$0.12**	**77.95 $\mp$0.06**	**70.39 $\mp$0.23**

Table 2: Top-1 and top-5 acc. on ImageNet. Teacher-Student **(a)**: RN34-RN18 **(b)**: RN50-MNV2. **Bold**: best in its category.

Setting		Teacher	Student	KD	DKD	QUEST	**letKD-1**	**letKD-2**
(a)	Top-1	73.31	69.75	70.66	71.70	71.67	72.33	**72.38**
	Top-5	91.42	89.07	89.88	90.41	90.67	91.06	**91.15**
(b)	Top-1	76.13	68.87	68.58	72.05	72.54	73.78	**73.98**
	Top-5	92.86	88.76	88.98	91.05	91.13	91.81	**92.00**

ImageNet(Tab. 2). We verify the scalability of our methods by performing experiments on large-scale datasets including ImageNet. Overall, our method letKD-2 outperforms all the other methods on all the different teacher-student combinations with homogeneous and heterogeneous architecture settings.

5.2 Ablations and Behavior Analysis

With and without K-means. To validate our analysis in § 4.2.1 that repurposes the 3x3 kernels of a 3x3-BN-ReLU-1x1 block as the cluster centers for the semantic entities, we evaluate our method on CIFAR-100 dataset with soft assignments $p_{\mathcal{T}}$ computed using the 3x3 kernels as explained in § 4.2.1 and the clusters obtained through K-means. We denote the experiments with (without) our KD layer as letKD-1 (QUEST) for both cases.

The results provided in Tab. 3 show that the exploitation of 3x3 kernels can replace K-means based supervision. We observe competitive and even better performances with 3x3 kernels as the cluster centers. Such results validate our claims suggesting

Table 3: Average top-1 accuracies on CIFAR-100 over 5 trials to validate the analysis on 3x3 kernels.

Clusters →	3x3 Kernels		K-Means		Teacher	Student
Archs. ↓	QUEST	letKD-1	QUEST	letKD-1		
RN56-RN20	71.92	72.11	71.84	72.44	72.34	69.06
RN110-RN32	74.31	74.44	74.08	74.40	74.31	71.14
RN83-RN29	72.41	72.61	72.48	73.33	73.84	70.53

that 3x3 kernels inherently learn templates (*i.e.*, cluster centers) that are capturing some semantic entities owing to the template matching paradigm introduced in § 4.1.

Let the student decide. We argue that using the soft labels obtained by K-means clustering of the teacher's features in the intermediate layers eventually results in pushing the student to imitate the teacher's geometry. As argued previously [31], the student can fail to effectively exploit that geometry due to capacity differences. Nonetheless, our KD layer is equipped with a mechanism to discard the information that may have a detrimental effect on the performance. To validate, we evaluate the multi-layer version of our method with K-means supervision in both layers, denoted as letKD-1 (2). We also evaluate QUEST under the same setting, denoted as QUEST (2). In the intermediate layer, we apply a temperature to the logits before obtaining $p_{\mathcal{T}}$ for both cases. The results in Tab. 4 on CIFAR-100 with RN56-RN20 show that QUEST with intermediate KD degrades the performance while our layer successfully shapes the knowledge to fit its representation capability.

Table 4: Effect of multi-layer distillation with QUEST and letKD-1.

Method	Top-1 Acc.
QUEST	71.84
QUEST (2)	71.79
letKD-1	72.44
letKD-1 (2)	72.73

Effect of the KD layer. Towards the understanding of the impact of our KD layer for intermediate layer (*i.e.*, lower level) supervision, we measured the classification capacity of the student trained with our methods. We achieved this by fitting a linear classifier to the global features of the trained student at that layer. Owing to this study with the results tabulated in the supplementary material [11, Tab. 4], we showed that our KD layer can improve the student's decision capacity significantly. To further validate our KD layer's ability to shape the intermediate features of the student by exploiting the teacher's knowledge, we analyzed the effect of how enhancing the student's features with the weighted combinations of the learned semantic vectors improves the overall performance. Namely, we set $\alpha = 0$ in (4.1) to lift the knowledge-based feature transform and compare its performance with $\alpha = 1$. The results presented in the supplementary material [11, Tab. 5] show consistent improvement of the inclusion of our KD layer. Finally, we addressed whether the performance increase is coming from the method or the capacity increase introduced by our KD layer. As summarized in the supplementary material [11, Tab. 6], we compared the performance of the three methods as FitNet, FitNet equipped with our KD layer at the penultimate layer with and without supervision to show that even though the capacity of the student is increased due to our KD layer, the major contribution for the performance occurs by combining it with our supervision.

6 Conclusion

We bring a different perspective to KD formulation in terms of a portable residual layer that improves KD by explicitly embedding the teacher's knowledge in feature transform. This way, we enable the student to discard nuisance information and feed forward transferred knowledge deeper for improved inference. To facilitate knowledge transfer in the intermediate layers, we also propose a novel form of supervision based on teacher's decisions. With extensive empirical studies, we validated the effectiveness of the proposed KD layer in various KD benchmarks.

References

[1] Sungsoo Ahn, Shell Xu Hu, Andreas Damianou, Neil D. Lawrence, and Zhenwen Dai. Variational information distillation for knowledge transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[2] Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? In *Advances in Neural Information Processing Systems*, volume 27, 2014.

[3] Defang Chen, Jian-Ping Mei, Hailin Zhang, Can Wang, Yan Feng, and Chun Chen. Knowledge distillation with the reused teacher classifier. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11933–11942, June 2022.

[4] Pengguang Chen, Shu Liu, Hengshuang Zhao, and Jiaya Jia. Distilling knowledge via knowledge review. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5008–5017, June 2021.

[5] Jiequan Cui, Pengguang Chen, Ruiyu Li, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Fast and practical neural architecture search. In *The IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6508–6517, 2019.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[7] Ada Gorgun, Yeti Z. Gurbuz, and Aydin Alatan. Feature embedding by template matching as a resnet block. In *33rd British Machine Vision Conference 2022, BMVC 2022, London, UK, November 21-24, 2022*. BMVA Press, 2022.

[8] Jianping Gou, Baosheng Yu, Stephen J. Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819, 2021.

[9] Yeti Z Gürbüz and A Aydın Alatan. A novel bovw mimicking end-to-end trainable cnn classification framework using optimal transport theory. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 3053–3057. IEEE, 2019.

[10] Yeti Z Gurbuz and A Aydin Alatan. Generalizable embeddings with cross-batch metric learning. *arXiv preprint arXiv:2307.07620*, 2023.

[11] Ada Görgün, Yeti Z. Gürbüz, and A. Aydın Alatan. Supplementary material for "knowledge distillation layer that lets the student decide". [https://github.com/adagorgun/letKD-framework], 2023.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016.

[14] Byeongho Heo, Minsik Lee, Sangdoo Yun, and Jin Young Choi. Knowledge transfer via distillation of activation boundaries formed by hidden neurons. In *AAAI Conference on Artificial Intelligence*, 2018.

[15] Byeongho Heo, Jeesoo Kim, Sangdoo Yun, Hyojin Park, Nojun Kwak, and Jin Young Choi. A comprehensive overhaul of feature distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

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

[17] A Hoffman, J Kruskal, and M Jünger. Introduction to integral boundary points of convex polyhedra. *Jünger M et al (eds)*, 50:1958–2008, 2010.

[18] Himalaya Jain, Spyros Gidaris, Nikos Komodakis, Patrick Pérez, and Matthieu Cord. Quest: Quantized embedding space for transferring knowledge. *European Conference on Computer Vision (ECCV)*, 2020.

[19] Alex Krizhevsky et al. Learning multiple layers of features from tiny images, 2009.

[20] Ya Le and Xuan S. Yang. Tiny imagenet visual recognition challenge, 2015.

[21] Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. Deeply-supervised nets. In *Artificial intelligence and statistics*, pages 562–570. PMLR, 2015.

[22] Yawei Li, Kamil Adamczewski, Wen Li, Shuhang Gu, Radu Timofte, and Luc Van Gool. Revisiting random channel pruning for neural network compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 191–201, June 2022.

[23] Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. Rethinking the value of network pruning. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJlnB3C5Ym.

[24] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: practical guidelines for efficient cnn architecture design. In *European Conference on Computer Vision (ECCV)*, pages 122–138, 2018.

[25] Seyed-Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, and Hassan Ghasemzadeh. Improved knowledge distillation via teacher assistant: Bridging the gap between student and teacher. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[26] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[27] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019.

[28] Baoyun Peng, Xiao Jin, Jiaheng Liu, Dongsheng Li, Yichao Wu, Yu Liu, Shunfeng Zhou, and Zhaoning Zhang. Correlation congruence for knowledge distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[29] 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.

[30] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018.

[31] Jie Song, Haofei Zhang, Xinchao Wang, Mengqi Xue, Ying Chen, Li Sun, Dacheng Tao, and Mingli Song. Tree-like decision distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13488–13497, June 2021.

[32] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[33] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. In *International Conference on Learning Representations*, 2020.

[34] Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[35] Kunran Xu, Lai Rui, Yishi Li, and Lin Gu. Feature normalized knowledge distillation for image classification. *European Conference on Computer Vision (ECCV)*, 2020.

[36] Kohei Yamamoto. Learnable companding quantization for accurate low-bit neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5029–5038, June 2021.

[37] Zhendong Yang, Zhe Li, Mingqi Shao, Dachuan Shi, Zehuan Yuan, and Chun Yuan. Masked generative distillation. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision – ECCV 2022*, pages 53–69, Cham, 2022. Springer Nature Switzerland. ISBN 978-3-031-20083-0.

[38] Han-Jia Ye, Su Lu, and De-Chuan Zhan. Distilling cross-task knowledge via relationship matching. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[39] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *British Machine Vision Conference 2016*. British Machine Vision Association, 2016.

[40] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *International Conference on Learning Representations*, 2017.

[41] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[42] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6848–6856, 2018.

[43] 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 (CVPR)*, pages 11953–11962, June 2022.

[44] Bolei Zhou, David Bau, Aude Oliva, and Antonio Torralba. Interpreting deep visual representations via network dissection. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[45] Yichen Zhu and Yi Wang. Student customized knowledge distillation: Bridging the gap between student and teacher. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5037–5046, 2021. doi: 10.1109/ICCV48922.2021.00501.