Teacher Features-Driven Regularization for Knowledge Distillation

Marco Orsingher*
Ambarella Inc.
morsingher@ambarella.com

Anthony Dell'Eva*
Ambarella Inc.
adelleva@ambarella.com

Yu-Ming Lee
Ambarella Inc.
ymlee@ambarella.com

Abstract

*Knowledge distillation (KD) aims at improving the accuracy of a small student network by mimicking the outputs or the intermediate features of a large teacher model. In this paper, we show that directly matching latent representations is suboptimal, since network layers are typically non-injective functions and different features at an early level might produce the same output at later levels. To this end, we introduce **TeaFeD**, a novel **Tea**cher **Fe**atures-**D**riven regularization loss for feature-based KD. Our intuition is to train each student layer to produce robust representations by feeding them to the corresponding subsequent teacher layers and matching the outputs with the teacher features at such level of hierarchy. This allows for knowledge transfer from a higher level of abstraction and enables the student to learn more generalizable features. Our method is extensively tested, analyzed and ablated on standard benchmarks across different vision tasks, showing state-of-the-art performances.*

1. Introduction

The widespread diffusion of mobile and wearable devices requires efficient network inference on hardware with limited resources. However, modern deep learning models benefit from large architectures [12, 18, 31, 45, 48] that fail to meet the computational and memory requirement of such devices. In order to bridge this gap, three main neural compression approaches have been proposed: distilling the knowledge from large networks to smaller models [21, 38, 53, 59], pruning redundant parameters [13, 16, 29, 40, 54] and quantizing the weights to low-precision formats [14].

Knowledge Distillation (KD) is a model compression technique in which a lightweight *student* network is trained to mimic the outputs [21, 33, 59] or the intermediate representations [22, 38, 46, 53] of a much larger *teacher* model. This allows the knowledge learned by the teacher to be effectively transferred to the student, thus achieving a higher accuracy than training it from scratch.

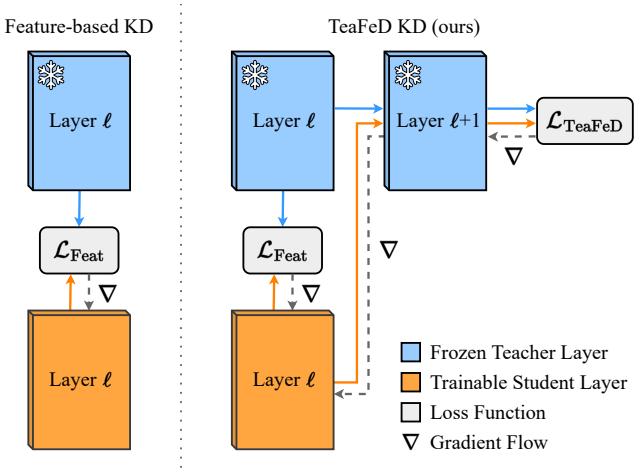


Figure 1. Overview of the proposed regularization loss for KD. We inject the student features at layer ℓ to the next *frozen* teacher layer $\ell + 1$ and match them with the corresponding teacher features.

Existing KD methods fall into two main categories: logit-based [21, 33, 59] and feature-based [22, 38, 46, 53]. KD was originally proposed in [21] for the classification task, as a method to match the output logits of two models with different capacities. The logits are converted to pseudo-probabilities with a softmax with rescaled temperature, and the KL divergence between the two distributions is minimized. Recently, feature-based methods have emerged by following the insights in [38]. The key idea is to align the intermediate representations of the student to those of the teacher, in order to boost the distillation performances.

However, directly matching the feature maps of the teacher is not a sufficient condition for the student to predict the correct output. Moreover, it is a hard optimization constraint to satisfy when the two models have different architectures or capacity. To this end, we propose *TeaFeD*, a novel *Tea*cher *Fe*atures-*D*riven regularization approach for KD, which is visualized in Fig. 1. The key idea is to train a generic student layer ℓ to produce a latent representation for the next teacher layer $\ell + 1$ that matches the corresponding

*Equal contribution.

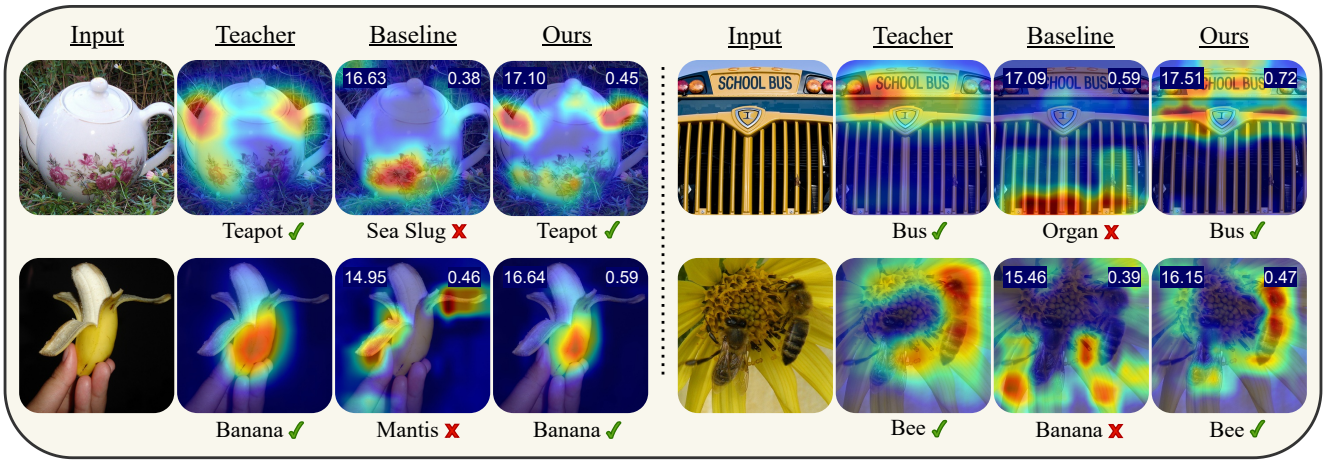


Figure 2. Visualization of the representations learned by the student with the baseline method [38] and our approach. The Euclidean and cosine distances, shown in the upper left and right corners of the students results, respectively, confirm that the baseline produces closer features to the teacher. However, it fails to classify inputs correctly as it focuses on irrelevant pixels, while our approach matches the teacher's attention. Activation maps are computed with Grad-CAM [41], while distances are computed on the last backbone feature vector.

teacher features. Since network layers typically implement non-injective functions, different features at level ℓ might produce the same output at level $\ell + 1$. Our approach allows for gradients to be propagated from a higher level of abstraction, thus providing more flexibility to the student in learning intermediate representations.

The effect of this regularization loss on the features learned by the student network is shown in Fig. 2. Even if the direct matching baseline [38] produces closer representations to the teacher, as determined by both Euclidean and cosine distances, it fails to classify correctly the input images. The activation maps, computed with Grad-CAM [41], prove that these errors stem from the focus on irrelevant pixels, while our approach allows the student to concentrate on the same image regions as the teacher. For example, the colorful teapot decoration and the school bus radiator grille are misinterpreted by the baseline as sea slug and organ, respectively, while our method yields the correct output attending to the relevant image region.

Therefore, our contributions to existing distillation literature can be summarized as follows:

- We propose a novel approach for feature-based KD that complements direct matching with a teacher features-driven regularization loss.
- We show that relaxing the feature matching constraint at corresponding layers by distilling knowledge from a higher level of abstraction enables the student model to learn robust and generalizable latent representations.
- We prove the effectiveness of our regularizer on standard benchmarks, showing state-of-the-art results for a wide variety of vision tasks, including image classification, semantic/instance segmentation and object detection.

2. Related Work

Logit-based KD. Logit-based methods distill the knowledge only at output level, by minimizing the divergence between the logits predicted by the teacher and the student model [21, 23, 33, 58, 59]. The seminal KD paper [21] proposed to rescale the temperature parameter in the last softmax layer, in order to smooth the output distribution and focus on negative logits [15]. Recently, DKD [59] showed that reformulating the distillation loss into a weighted sum for positive and negative logits effectively decouples their contributions and further boosts performances. On the other hand, DML [58] employed a mutual learning paradigm to train both models simultaneously, while TAKD [33] introduced an intermediate teacher assistant to reduce the capacity gap between the teacher and the student. Finally, Lin *et al*. [23] presented a multi-level framework to distill logits across instances, batches and categories. Logit-based methods are efficient in terms of both memory and compute, but they typically achieve lower results with respect to state-of-the-art feature-based approaches.

Feature-based KD. Feature-based methods aim at transferring intermediate representations from the teacher to the student, in order to provide a richer supervision [1–4, 17, 20, 22, 30, 36, 38, 43, 46, 47, 50, 52, 53, 56]. Following the idea presented in [38] to mimic the teacher features via a simple L_2 loss, several approaches have been developed to improve different aspects of the pipeline. Some works [22, 52, 53] proposed to mask the feature maps, thus allowing the student to focus on relevant parts of the image. Popular masking strategies include learnable tokens [22], random masks reconstructed with generative layers [53],

and fixed separation of background and foreground for object detection [52]. Another approach to alleviate the constraint of direct feature matching is to transform the raw features into attention maps [17, 43, 52, 56] or to compute local statistics, such as pairwise similarities [47] and the Pearson correlation coefficient [2]. Furthermore, cross-level distillation has been explored in [3, 36], with the goal of sharing features across different network layers, thus integrating knowledge from multiple learning stages. Inspired by these methods, we propose a straightforward approach to complement direct feature matching with a novel regularization term that forces each student layer to learn meaningful features from the subsequent teacher layers.

3. Method

3.1. Preliminaries

Logit-based KD. The fundamental baseline in KD research is the vanilla logit-based method introduced in [21]. Given a model trained on a classification task, the output logits $\mathbf{z} \in \mathbb{R}^C$ can be converted to pseudo-probabilities using the softmax activation as:

$$y_j = \frac{e^{z_j/t}}{\sum_{c=1}^{C} e^{z_c/t}}, \tag{1}$$

where z_j and y_j are the logit and probability values for j-th category, respectively. In the KD context, the *temperature* parameter t is often larger than 1, producing softer distributions which alleviate the over-confidence problem in neural networks [15] ($t = 1$ yields the vanilla softmax output). To transfer knowledge from complex to compact models, the original logit-based technique minimizes the Kullback-Leibler (KL) divergence between teacher and student outputs as:

$$\mathcal{L}_{\text{Log}} = \text{KL}\left(\mathbf{y}^T || \mathbf{y}^S\right) = \sum_{j=1}^{C} y_j^T \log\left(\frac{y_j^T}{y_j^S}\right), \tag{2}$$

where the superscripts T and S indicate whether the outputs originate from the teacher or the student, respectively. As a result, the *dark knowledge* is transferred via soft labels from the teacher, improving the prediction of the student.

Feature-based KD. The distillation approaches based on features are versatile and mostly agnostic with respect to the model architecture, which can vary considerably for different tasks. The basic feature-based method [38] can be formulated as:

$$\mathcal{L}_{\text{Feat}}^{\ell} = \|F_\ell^T - \mathcal{A}_\ell(F_\ell^S)\|_2^2. \tag{3}$$

Here, $F_\ell^T \in \mathbb{R}^{C_{T_\ell} \times H_{T_\ell} \times W_{T_\ell}}$ and $F_\ell^S \in \mathbb{R}^{C_{S_\ell} \times H_{S_\ell} \times W_{S_\ell}}$ correspond to the teacher and student feature maps extracted from layer ℓ, while $\mathcal{A}_\ell$ denotes the convolutional adaptation layer needed to align the student features dimensions with those of the teacher.

Algorithm 1 TeaFeD Regularization in PyTorch

```
# (img, y):  Input and Label
# (S, T): Student and Teacher
# num_layers:  Number of Layers
# [adapt]:  Adaptation Layers
# (g1, g2, a, b):  Loss Hyperparameters
import torch.nn.functional as F
y_S, feat_S = S(img)
y_T, feat_T = T(img)
L_task = F.cross_entropy(y_S, y)
L_log = F.kl_div(y_S, y_T)
L_feat = 0.
L_teafed = 0.
for l in range(num_layers):
  F_S = adapt[l](feat_S[l])
  l2_feat = F.mse_loss(F_S, feat_T[l])
  L_feat += (g2 ** -l) * l2_feat
              l2_teafed = F.mse_loss(T[l+1](F_S),
feat_T[l+1])
  L_teafed += (g1 ** -l) * l2_teafed
L = L_task + L_log + a * L_feat + b * L_teafed
```

The approach forces the student to directly imitate the teacher features. This may not be optimal if the models' architectures diverge significantly or their capacities are widely different, given the student's inability to exactly replicate the more complex features. To address the issue, we introduce a novel feature-based regularization, which employs a straightforward L_2 loss to steer the student toward a more robust solution space.

3.2. Teacher Features-Driven Regularization

Given a teacher network with a structured hierarchy of layers (or stages) denoted as $\{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_L\}$, and a parallel student network denoted as $\{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_L\}$, our TeaFeD method regularizes the learning process at matching layers. At each stage ℓ, the aligned student features $\mathcal{A}_\ell(F_\ell^S)$ are propagated to the corresponding subsequent frozen teacher stage $\mathcal{T}_{\ell+1}$, which provides new features on a higher level of abstraction. Thus, the loss is formulated as:

$$\mathcal{L}_{\text{TeaFeD}}^{\ell} = \|F_{\ell+1}^T - \mathcal{T}_{\ell+1}\left(\mathcal{A}_\ell(F_\ell^S)\right)\|_2^2. \tag{4}$$

Due to non-linear mappings and pooling transformations, the teacher layer implementing the loss function is typically non-injective. This suggests that different inputs might produce identical latent representations, allowing for more flexibility in the features derived by the student. Therefore, our penalization term effectively acts as a powerful regularizer, promoting better generalization.

Overall loss. We employ a simple exponentially increasing weighting scheme on the sequence of features for both our loss and the vanilla feature-based loss. The aggregated

Architecture	Homogeneous				Heterogeneous			
Student	ResNet-116 73.70	VGG-8 71.00	ResNet-8x4 73.30	WRN-40-1 72.17	VGG-8 71.00	ShuffleNetV1 71.15	MobileNetV2 65.94	ShuffleNetV2 73.33
KD [21]	76.26±0.26	73.43±0.23	74.66±0.12	74.12±0.21	72.51±0.21	75.27±0.08	69.35±0.42	75.84±0.22
FitNet [38]	76.48±0.31	73.54±0.12	74.46±0.21	73.86±0.08	72.82±0.29	75.36±0.33	69.49±0.19	75.91±0.13
SP [47]	76.20±0.37	73.38±0.23	74.19±0.32	73.98±0.28	73.35±0.19	75.56±0.23	68.89±0.19	76.29±0.19
VID [1]	76.47±0.29	73.65±0.18	74.87±0.03	74.02±0.14	73.36±0.08	74.87±0.55	68.94±0.36	76.03±0.22
CRD [46]	76.83±0.26	74.37±0.14	75.72±0.39	74.44±0.16	73.80±0.07	75.88±0.24	70.05±0.45	76.40±0.24
SRRL [50]	76.79±0.40	73.85±0.07	75.51±0.21	74.78±0.23	73.49±0.12	75.59±0.14	69.96±0.72	76.19±0.10
SemCKD [3]	77.07±0.55	74.35±0.11	76.00±0.22	74.39±0.39	75.33±0.31	75.78±0.34	70.26±0.49	77.97±0.26
ReviewKD [36]	77.42±0.03	73.11±0.12	75.87±0.14	**75.35±0.17**	73.41±0.08	75.49±0.19	70.25±0.24	77.55±0.25
CAT-KD [17]	77.39±0.18	74.46±0.10	76.48±0.11	75.10±0.19	75.66±0.28	75.43±0.24	69.84±0.31	77.01±0.16
Ours	**77.54±0.13**	**75.01±0.06**	**76.64±0.16**	75.18±0.22	**75.74±0.14**	**76.36±0.17**	**70.94±0.11**	**78.84±0.21**
Teacher	ResNet-110x2 78.17	VGG-13 75.14	ResNet-32x4 79.21	WRN-40-2 76.53	ResNet-32x4 79.21	VGG-13 75.14	WRN-40-2 76.53	ResNet-32x4 79.21

Table 1. Top-1 test accuracy (%) of homogeneous and heterogeneous teacher-student pairs on CIFAR-100. Average over 4 runs.

Architecture	Homogeneous		Heterogeneous	
Student	VGG-8 55.83	ResNet-8x4 55.53	ShuffleNetV1 58.64	VGG-8 55.83
KD [21]	60.36±0.22	58.16±0.20	61.25±0.52	60.52±0.42
FitNet [38]	60.35±0.23	58.21±0.10	61.47±0.23	60.69±0.16
SP [47]	60.22±0.07	58.71±0.18	61.53±0.06	60.57±0.31
VID [1]	60.81±0.08	58.61±0.19	61.55±0.19	61.16±0.06
CRD [46]	61.72±0.22	59.62±0.12	63.50±0.02	61.81±0.09
SRRL [50]	61.41±0.16	59.72±0.29	62.74±0.64	62.02±0.18
SemCKD [3]	61.40±0.18	57.50±0.57	62.53±0.77	62.57±0.59
ReviewKD [36]	60.01±0.06	59.83±0.24	63.92±0.47	61.86±0.34
CAT-KD [17]	61.52±0.08	58.13±0.29	63.70±0.04	62.59±0.19
Ours	**62.24±0.07**	**60.28±0.26**	**64.21±0.27**	**63.75±0.26**
Teacher	VGG-13 61.34	ResNet-32x4 64.53	VGG-13 61.34	ResNet-32x4 64.53

Table 2. Top-1 test accuracy (%) of homogeneous and heterogeneous teacher-student pairs on TinyImageNet. The results are averaged over 3 runs.

losses can be expressed as, respectively:

$$\mathcal{L}_{\text{TeaFeD}} = \sum_{\ell=1}^{L} \gamma_1^{-\ell} \, \mathcal{L}_{\text{TeaFeD}}^{\ell}, \tag{5}$$

$$\mathcal{L}_{\text{Feat}} = \sum_{\ell=1}^{L} \gamma_2^{-\ell} \, \mathcal{L}_{\text{Feat}}^{\ell}. \tag{6}$$

Student models are trained with the cumulative loss:

$$\mathcal{L} = \mathcal{L}_{\text{Task}} + \mathcal{L}_{\text{Log}} + \alpha \, \mathcal{L}_{\text{Feat}} + \beta \, \mathcal{L}_{\text{TeaFeD}}, \tag{7}$$

where α and β are hyperparameters to balance the losses and $\mathcal{L}_{\text{Task}}$ represents the task-specific loss (e.g., for classification, the cross-entropy computed between student outputs and ground truth labels). In practice, we set $\alpha = \beta$ and $\gamma_1 = \gamma_2 = \gamma \in (0, 1]$ to halve the number of hyperparameters to be tuned. The core steps of our approach are delineated in Algorithm 1.

4. Experiments

4.1. Classification

Datasets and baselines. We employ two standard image classification datasets for our experiments. CIFAR-100 [24] consists of 60k images (50k for training and 10k for validation) from 100 classes, with an image resolution of 32×32. TinyImageNet [25] is a subset of ImageNet [10] and comprises 110k images from 200 categories, divided into 100k for training and 10k for validation, with resolution 64×64. In addition to the reference KD method [21], we reproduce various state-of-the-art strategies for comparison, including FitNet [38], SP [47], VID [1], CRD [46], SRRL[50], SemCKD [3], ReviewKD [36], and CAT-KD [17]. Note that all the approaches, except for KD itself, incorporate the vanilla KD loss (Eq. 2).

Training settings. Following [18, 55], we employ standard data augmentation and normalize all images based on channel means and standard deviations. For CIFAR-100, the total training epochs are 240, the batch size is set to 64, and the base learning rate for SGD optimizer to 0.05. The hyperparameters for our loss are configured as $\{\alpha = 20, \gamma = 0.8\}$. For TinyImageNet, the models are trained for 200 epochs, the batch size is 128 and the base learning rate is 0.1, while the settings for the loss parameters are $\{\alpha = 5, \gamma = 0.8\}$ [1]. For classification, distillation is performed on the backbone features. The ablation study on losses and features employed can be found in Sec. 4.4.

Results. The classification results are shown in Tab. 1 and Tab. 2, with evaluations based on Top-1 accuracy. It can be noticed that we consistently outperform other state-of-the-art approaches for nearly every pair on both datasets, includ-

[1] Refer to the supplementary materials for further reproducibility.

	Teacher	Student	mAP	AP_S	AP_M	AP_L
One-stage	RetinaNet ResNeXt101 41.0	RetinaNet-Res50	37.4	20.6	40.7	49.7
		FKD [42]	39.6	22.7	43.3	52.5
		CWD [43]	40.8	22.7	44.5	55.3
		FGD [52]	40.7	22.9	45.0	54.7
		MGD [53]	41.0	23.4	45.3	**55.7**
		MasKD [22]	41.0	22.6	45.3	55.3
		Ours	**41.3**	**23.4**	**45.9**	54.6
Anchor-free	RepPoints ResNeXt101 44.2	RepPoints-Res50	38.6	22.5	42.2	50.4
		FKD [42]	40.6	23.4	44.6	53.0
		CWD [43]	42.0	24.1	46.1	55.0
		FGD [52]	42.0	24.0	45.7	55.6
		MGD [53]	42.3	24.4	46.2	55.9
		MasKD [22]	42.5	**24.9**	46.1	**56.8**
		Ours	**42.9**	24.8	**47.1**	56.3
Two-stage	Cascade Mask RCNN ResNeXt101 47.3	Faster RCNN-Res50	38.4	21.5	42.1	50.3
		FKD [42]	41.5	23.5	45.0	55.3
		CWD [43]	41.7	23.3	45.5	55.5
		FGD [52]	42.0	23.8	46.4	55.5
		MGD [53]	42.1	23.7	46.4	56.1
		MasKD [22]	42.4	24.2	**46.7**	55.9
		Ours	**42.4**	**24.4**	46.3	**56.4**

Table 3. Object detection results on COCO.

	Teacher	Student	mAP	AP_S	AP_M	AP_L
One-stage	SOLO-Res101 3x, MS 37.1	SOLO-Res50-1x	33.1	12.2	36.1	50.8
		FGD [52]	36.0	14.5	39.5	54.5
		MGD [53]	36.2	14.2	39.7	**55.3**
		Ours	**36.4**	**15.3**	**40.1**	55.2
Two-stage	Cascade Mask RCNN ResNeXt101 41.1	Mask RCNN-Res50	35.4	16.6	38.2	52.5
		FGD [52]	37.8	17.1	40.7	56.0
		MGD [53]	38.1	17.1	**41.1**	56.3
		Ours	**38.1**	**17.8**	41.0	**56.5**

Table 4. Instance segmentation results on COCO.

ing ReviewKD [36] and CAT-KD [17] which require careful hyperparameter tuning for each teacher-student combination. Remarkably, we improve accuracy by a considerable margin in several cases. For example, with the ShuffleNetV2/ResNet-32x4 pair on CIFAR-100, we achieve a 1% accuracy increase compared to the second-best method. Similarly, in the VGG-8/VGG-13 combination on TinyImageNet, the student surprisingly exceeds even the teacher performance. The outcomes prove that the introduced regularization term facilitates flexible feature learning, enhancing the student robustness regardless of dataset and architecture types.

4.2. Object Detection and Instance Segmentation

Dataset and baselines. The experiments are conducted on COCO2017 dataset [27], which comprises 120k training images and 5k validation images divided in 80 object categories. We compare our approach with recent distillation methods for detectors, including FKD [42], CWD [43], FGD [52], MGD [53], and MasKD [22].

Architecture	Homogeneous	Heterogeneous
Student	PspNet-Res18 69.85	DeepLabV3-Res18 73.20
SKDS [30]	72.70	73.87
CWD [43]	73.53	75.93
MGD [53]	74.10	76.31
Ours	**74.87**	**76.44**
Teacher	PspNet-Res101 78.34	PspNet-Res101 78.34

Table 5. Semantic segmentation results on Cityscapes.

Training settings. We train the models for 24 epochs with SGD optimizer (momentum = 0.9, weight decay = 10^{-4}). When the student and teacher share the same head structure, we adopt the inheriting strategy [52, 53] to initialize the student with the teacher's neck and head parameters. The hyperparameters are $\{\alpha = 5 \times 10^{-5}, \gamma = 1\}$ for one-stage detectors and $\{\alpha = 5 \times 10^{-7}, \gamma = 1\}$ for two-stage detectors. The distillation loss is computed on the feature maps extracted from the neck. The experiments are executed with MMDetection [5].

Results. The experiments for the object detection task include three different models, namely the one-stage detector RetinaNet [28], the anchor-free one-stage detector RepPoints [51] and the two-stage detector Faster RCNN [37]. The evaluation metrics are the mean bounding box Average Precision (mAP) and the variants AP_S, AP_M, and AP_L, corresponding to small, medium and large object sizes, respectively. In the context of instance segmentation, we focus on SOLO [49] and Mask RCNN [19] models, which are evaluated with the mean mask Average Precision metric (mAP). From the results presented in Tab. 3 and Tab. 4, it can be noticed that our method achieves better performances with respect to the other state-of-the-art approaches for both tasks. For instance, our TeaFeD regularization boosts significantly the RepPoints and Mask RCNN students, leading to 4.3 and 2.7 mAP improvement, respectively. Additionally, RetinaNet and SOLO obtain a remarkable 2.8 and 3.1 improvement on the AP_S metric.

4.3. Semantic Segmentation

Dataset and baselines. Semantic segmentation is evaluated on Cityscapes dataset [9], consisting of 5k urban scenes images divided in 3k for training, 500 for validation and 1.5k for testing. We compare our method with recent approaches, such as SKDS [30], CWD [43] and MGD [53].

Training settings. We train the student models with SGD (momentum = 0.9, weight decay = 5×10^{-4}) for 128 epochs

Architecture	Homogeneous	Heterogeneous
Student	VGG-8 71.00	ShuffleNetV2 73.33
$\ell \in \{1,2,3\}$	74.55±0.12	78.79±0.14
$\ell \in \{1,2\}$	74.49±0.15	78.64±0.10
$\ell = 4$	73.41±0.14	75.74±0.28
$\ell = 3$	73.53±0.14	76.12±0.07
$\ell = 2$	74.11±0.31	77.81±0.23
$\ell = 1$	74.21±0.13	77.27±0.21
w/o $\mathcal{L}_{\text{Log}}$	74.53±0.28	78.73±0.37
w/o $\mathcal{L}_{\text{Feat}}$	74.70±0.34	78.09±0.10
w/o $\mathcal{L}_{\text{TeaFeD}}$	74.49±0.13	78.07±0.19
Ours	**75.01±0.06**	**78.84±0.21**
Teacher	VGG-13 75.14	ResNet-32x4 79.21

Table 6. Ablation studies on different versions of the student for both homogeneous and heterogeneous settings. Layers are numbered in descending order. Ours is equivalent to $\ell \in \{1,2,3,4\}$. Results are averaged over 3 runs.

with batch size equal to 16. Following [53], our loss is computed only on the last backbone feature and the corresponding parameters are set to $\{\alpha = 2 \times 10^{-5}, \gamma = 1\}$. The experiments are conducted with MMSegmentation [8].

Results. We evaluate the segmentation task on both homogeneous and heterogeneous distillation settings, using mean Intersection over Union (mIoU) as the quantitative metric. Specifically, we employ PspNet-Res101 [60] as the teacher, while PspNet-Res18 [60] and DeepLabV3-Res18 [6] as students. The results in Tab. 5 confirm that our method is the best among the state-of-the-art approaches. In detail, the two students register a 5.02 and 3.24 improvement with respect to the original model trained without distillation, respectively.

4.4. Ablations

We conduct several ablation studies to assess the contribution of distinct layers and losses on the model predictions. The results in Tab. 6 are obtained by training a homogeneous and a heterogeneous classification network pair on CIFAR-100 with different configurations.

In order to evaluate the influence of layers/blocks, we perform distillation removing them sequentially. Note that, in this context, layers are enumerated from highest to lowest, with $\ell = 1$ corresponding to the last layer of the backbone. Two observations emerge from the accuracy values: firstly, integrating more layers is beneficial for the student, as demonstrated by training the model with only a limited

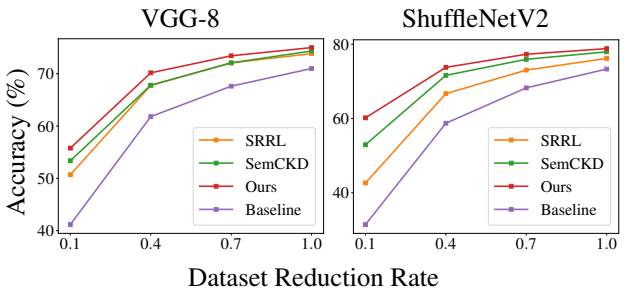


Figure 3. Analysis on the data efficiency of different KD methods when trained with different amounts of data on CIFAR-100 classification.

subset; secondly, the final layers contribute more to the network performance, as highlighted by the improved numerical evaluations at $\ell = 1$ and $\ell = 2$. This justifies the choice of applying exponential weighting to the feature-based losses, emphasizing the terminal layers.

Moreover, we investigate the impact of the different losses on the student effectiveness. The last rows of Tab. 6 confirm that our regularization objective plays a crucial role in the distillation process. Notably, removing our term leads to the most significant performance drop compared to the exclusion of the other distillation losses. Additionally, the results show that, when evaluated independently, our loss outperforms the vanilla feature-based approach. Nonetheless, the best accuracy is achieved through joint training.

4.5. Analysis

In this section, we provide an extensive analysis of our KD approach in terms of data and training efficiency, transferability of the features learned by the student, sensitivity to hyperparameters and impact of the adaptation layer. All the experiments are conducted for the classification task on CIFAR-100 [24] for a ShuffleNetV2 student distilled from a ResNet-32x4 teacher, except where stated otherwise.

Data Efficiency. We evaluate the dependency of different distillation approaches [3, 50] on the amount of training data, by reducing the training set of CIFAR-100 [24] to various ratios. Fig. 3 shows that we consistently outperform other methods across the whole spectrum of dataset reductions, for both ShuffleNetV2 and VGG-8 (with VGG-13 as teacher) architectures. This result proves that the proposed regularization loss can transfer knowledge from the teacher more efficiently.

Training Efficiency. While adding our $\mathcal{L}_{\text{TeaFeD}}$ as a learning objective leads to improved classification accuracy, it also increases the overall training time of the student network. We study this aspect by structurally pruning [13]

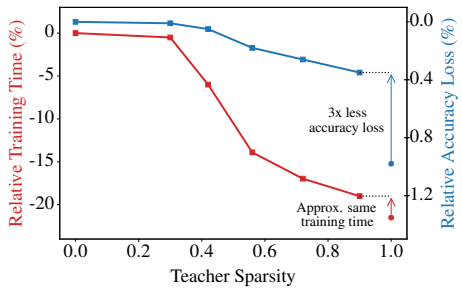


Figure 4. Analysis on how pruning the teacher allows to reduce the training time of our method almost to the baseline, while keeping much more accuracy. Results are averaged over 3 runs.

Architecture	Homogeneous		Heterogeneous	
Teacher	ResNet-110x2		VGG-13	
Student	ResNet-116		ShuffleNetV1	
CIFAR-100 →	STL-10	TinyImg	STL-10	TinyImg
KD [21]	63.67	29.59	67.70	36.00
FitNet [38]	64.22	30.64	67.94	36.30
SP [47]	64.90	30.93	68.16	36.94
VID [1]	64.73	30.16	68.29	35.67
CRD [46]	66.11	32.69	69.03	38.66
SRRL [50]	64.22	31.59	69.15	37.58
SemCKD [3]	66.42	34.32	69.17	38.63
ReviewKD [36]	66.54	**34.70**	68.50	37.79
CAT-KD [17]	65.73	32.75	70.82	39.36
Ours	**66.68**	34.11	**70.86**	**39.52**

Table 7. Features transferability from CIFAR-100 to other classification datasets. A linear classifier is trained on features extracted from a frozen student distilled with different KD methods.

the teacher at multiple sparsity values and analyzing the accuracy-efficiency tradeoff. Fig. 4 compares our results with the baseline feature matching approach [38]. It can be seen that distilling from a lighter teacher with our KD method allows to achieve approximately the same training time, while losing only 0.35% accuracy, which is almost $3\times$ less than the baseline 0.98% accuracy loss. Moreover, all the intermediate sparsity levels provide a substantial efficiency gain with a bounded performance loss, allowing to adapt the method to custom computational constraints.

Features Transferability. As we are interested in which features are learned by the student during the distillation procedure, we also analyze how such features transfer to datasets unseen at training time. To this end, the student trained on CIFAR-100 [24] is used as a frozen representation extractor for images from STL-10 [7] and TinyImageNet [25]. Then, we train a linear classifier on these representations and evaluate the accuracy on both datasets. Tab. 7 shows the results against multiple baselines for homogeneous and heterogeneous teacher-student pairs. The regularization effect introduced by our proposed loss allows the student to learn more transferable features with respect to concurrent KD methods, thus proving its effectiveness.

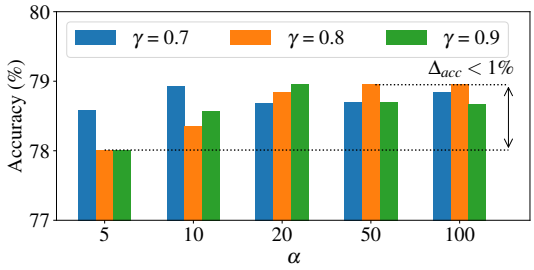


Figure 5. Analysis on the effect of varying the hyperparameters required to tune our regularization loss. Average over 3 runs.

Hyperparameters Sensitivity. The proposed approach depends on two main hyperparameters, namely the weights $\gamma \in (0, 1]$ of each feature layer and $\alpha \in \mathbb{R}$ for the feature-based KD losses. We show in Fig. 5 that the performances of our regularization term are minimally affected by the choice of such parameters. By letting $\gamma \in \{0.7, 0.8, 0.9\}$ and $\alpha \in \{5, 10, 20, 50, 100\}$, the accuracy difference between the best and worst configuration is less than 1%. Moreover, note we avoid cherry-picking a specific set of good hyperparameters, as our choice in Sec. 4.1, i.e. $\{\alpha = 20, \gamma = 0.8\}$ is not even the optimal one.

4.6. Sparse Self-Distillation

The proposed regularization loss allows to train an efficient student model with comparable accuracy with respect to the teacher network. To further enhance the deployment performances of the student, we also present a neural compression strategy, called *sparse self-distillation*, which integrates the distillation process in the pruning and fine-tuning loop. There are two main paradigms when pruning redundant parameters from a given model: (i) *unstructured* approaches [11, 16, 26, 34, 40] directly zero out individual weights based on their magnitude, without altering the network architecture, while (ii) *structured* [13, 29, 54] methods physically remove entire channels or layers. Our procedure can be applied with remarkable results to both strategies.

Procedure. Let S be the student network after KD training, $\mathcal{P}_r$ a generic pruning algorithm [13, 16, 40] with sparsity r and $\hat{S}_r = \mathcal{P}_r(S)$ the resulting model when applying $\mathcal{P}_r$ to S. Typically, $\hat{S}_r$ is fine-tuned for a few epochs with ground truth data and the task-specific loss $\mathcal{L}_{\text{Task}}(\hat{S}_r)$, in order to partially retrieve the accuracy lost during pruning. However, in practice, there is a tradeoff between the target sparsity and the bound on accuracy loss, especially for small

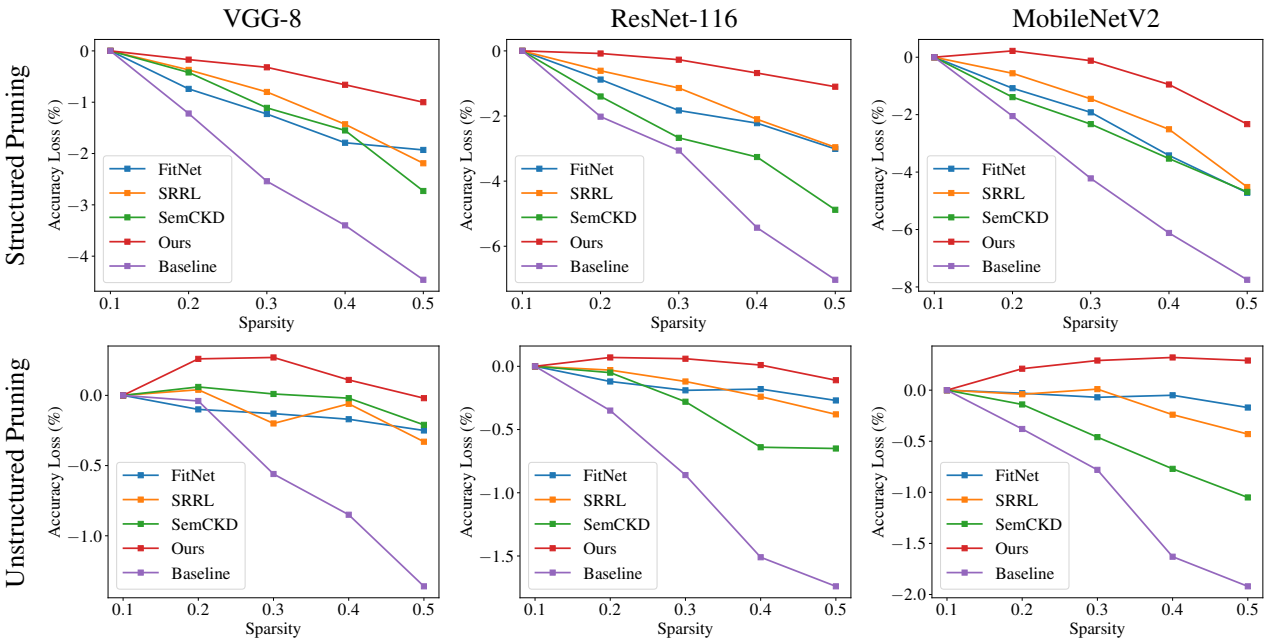


Figure 6. Quantitative results of our sparse self-distillation method for image classification on CIFAR-100. The baseline pruning and fine-tuning approach (without KD) is consistently outperformed by integrating KD in the loop for all networks, pruning and distillation algorithms. Moreover, our TeaFeD regularization loss provides the lowest accuracy loss across the whole sparsity spectrum.

networks already compressed with KD. We propose to alleviate this constraint by setting the original unpruned student S as the *teacher* and to self-distill it during the pruning process. This means that $\hat{S}_r$ is fine-tuned with the sum of two loss terms:

$$\mathcal{L} = \mathcal{L}_{\text{Task}}(\hat{S}_r) + \mathcal{L}_{\text{KD}}(\hat{S}_r, S) \tag{8}$$

The distillation loss $\mathcal{L}_{\text{KD}}$ forces the sparse student $\hat{S}_r$ to follow the output distribution and the learned representations of the original dense student S. In this way, the sparsity r can be pushed further, leading to the deployment of smaller and faster models with competitive accuracy. Note that this strategy is agnostic to the distillation method implemented by $\mathcal{L}_{\text{KD}}$. However, our regularization loss is particularly effective in this setting, for both structured and unstructured pruning algorithms.

Settings. We test our sparse self-distillation procedure on the CIFAR-100 dataset [24], with different network architectures, including VGG-8, ResNet-116 and MobileNetV2, as well as various distillation algorithms [3, 38, 50]. We also compare the proposed method against the baseline approach of iterative pruning and fine-tuning on ground truth data, without KD. Unstructured pruning is applied with the built-in PyTorch utility, while we rely on the state-of-the-art DepGraph [13] algorithm for structured pruning. In all the experiments, the remaining parameters of the pruned network $\hat{S}_r$ are fine-tuned for 50 epochs.

Results. The sparse self-distillation results are shown in Fig. 6, where two key results emerge. Firstly, integrating KD within the pruning and fine-tuning loop is consistently better than the baseline approach, for any distillation method. Secondly, our proposed TeaFeD regularization is particularly effective in this setting, outperforming all the other KD losses. Such results find evidence across different architectures and pruning algorithms. This allows to push the network sparsity further, with a much smaller accuracy loss, thus leading to faster models for the inference stage.

5. Conclusion

In this paper, we show that feature-based knowledge distillation can be improved by complementing the direct feature matching baseline with a teacher features-driven regularization loss, which enables the student model to learn robust intermediate representations. We extensively evaluate and ablate the proposed approach, outperforming state-of-the-art distillation methods on several vision tasks and benchmarks. Furthermore, we also introduce a novel neural compression strategy that effectively combines KD with network pruning, in order to improve the sparsity-accuracy tradeoff when deploying the student model on inference devices. The main limitation of our approach is the need for accessing the teacher weights at training time. As future work, we plan to tackle this limitation in order to support more general black-box teachers and to apply our method on non-convolutional architectures.

References

[1] Sungsoo Ahn, Shell Xu Hu, Andreas C. Damianou, Neil D. Lawrence, and Zhenwen Dai. Variational information distillation for knowledge transfer. In *CVPR*, pages 9163–9171. Computer Vision Foundation / IEEE, 2019. 2, 4, 7

[2] Weihan Cao, Yifan Zhang, Jianfei Gao, Anda Cheng, Ke Cheng, and Jian Cheng. Pkd: General distillation framework for object detectors via pearson correlation coefficient. *Advances in Neural Information Processing Systems*, 35: 15394–15406, 2022. 3

[3] Defang Chen, Jian-Ping Mei, Yuan Zhang, Can Wang, Zhe Wang, Yan Feng, and Chun Chen. Cross-layer distillation with semantic calibration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7028–7036, 2021. 3, 4, 6, 7, 8, 12

[4] 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*, pages 11933–11942, 2022. 2, 12

[5] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 5

[6] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017. 6

[7] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223. JMLR Workshop and Conference Proceedings, 2011. 7

[8] MMSegmentation Contributors. MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark. https://github.com/open-mmlab/mmsegmentation, 2020. 6

[9] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016. 5

[10] 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. Ieee, 2009. 4

[11] Xin Dong, Shangyu Chen, and Sinno Pan. Learning to prune deep neural networks via layer-wise optimal brain surgeon. *Advances in neural information processing systems*, 30, 2017. 7

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 1

[13] Gongfan Fang, Xinyin Ma, Mingli Song, Michael Bi Mi, and Xinchao Wang. Depgraph: Towards any structural pruning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16091–16101, 2023. 1, 6, 7, 8

[14] Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W Mahoney, and Kurt Keutzer. A survey of quantization methods for efficient neural network inference. *arXiv preprint arXiv:2103.13630*, 2021. 1

[15] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR, 2017. 2, 3

[16] Yiwen Guo, Anbang Yao, and Yurong Chen. Dynamic network surgery for efficient dnns. *Advances in neural information processing systems*, 29, 2016. 1, 7

[17] Ziyao Guo, Haonan Yan, Hui Li, and Xiaodong Lin. Class attention transfer based knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11868–11877, 2023. 2, 3, 4, 5, 7

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1, 4, 12

[19] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 5

[20] 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*, pages 1921–1930, 2019. 2

[21] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*, 2015. 1, 2, 3, 4, 7

[22] Tao Huang, Yuan Zhang, Shan You, Fei Wang, Chen Qian, Jian Cao, and Chang Xu. Masked distillation with receptive tokens. In *The Eleventh International Conference on Learning Representations*, 2023. 1, 2, 5

[23] Ying Jin, Jiaqi Wang, and Dahua Lin. Multi-level logit distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24276–24285, 2023. 2

[24] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 4, 6, 7, 8

[25] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015. 4, 7

[26] Namhoon Lee, Thalaiyasingam Ajanthan, Stephen Gould, and Philip H. S. Torr. A signal propagation perspective for pruning neural networks at initialization. In *ICLR*, 2020. 7

[27] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In *ECCV*, pages 740–755. Springer, 2014. 5

[28] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 5

[29] Liyang Liu, Shilong Zhang, Zhanghui Kuang, Aojun Zhou, Jing-Hao Xue, Xinjiang Wang, Yimin Chen, Wenming Yang, Qingmin Liao, and Wayne Zhang. Group fisher pruning for practical network compression. In *International Conference on Machine Learning*, pages 7021–7032. PMLR, 2021. 1, 7

[30] Yifan Liu, Ke Chen, Chris Liu, Zengchang Qin, Zhenbo Luo, and Jingdong Wang. Structured knowledge distillation for semantic segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2604–2613, 2019. 2, 5

[31] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11976–11986, 2022. 1

[32] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet V2: practical guidelines for efficient CNN architecture design. In *ECCV (14)*, pages 122–138. Springer, 2018. 12

[33] Seyed Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, Nir Levine, Akihiro Matsukawa, and Hassan Ghasemzadeh. Improved knowledge distillation via teacher assistant. In *Proceedings of the AAAI conference on artificial intelligence*, pages 5191–5198, 2020. 1, 2

[34] Sejun Park, Jaeho Lee, Sangwoo Mo, and Jinwoo Shin. Lookahead: A far-sighted alternative of magnitude-based pruning. In *ICLR*, 2020. 7

[35] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. 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. In *NeurIPS*, pages 8024–8035, 2019. 12

[36] Chen Pengguang, Liu Shu, Zhao Hengshuang, and Jiaya Jia. Distilling knowledge via knowledge review. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2, 3, 4, 5, 7

[37] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015. 5

[38] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 1, 2, 3, 4, 7, 8, 13, 14

[39] Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, pages 4510–4520. Computer Vision Foundation / IEEE Computer Society, 2018. 12

[40] Victor Sanh, Thomas Wolf, and Alexander Rush. Movement pruning: Adaptive sparsity by fine-tuning. *Advances in Neural Information Processing Systems*, 33:20378–20389, 2020. 1, 7

[41] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017. 2, 13, 14

[42] Zhiqiang Shen and Eric Xing. A fast knowledge distillation framework for visual recognition. In *European Conference on Computer Vision*, pages 673–690. Springer, 2022. 5

[43] Changyong Shu, Yifan Liu, Jianfei Gao, Zheng Yan, and Chunhua Shen. Channel-wise knowledge distillation for dense prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5311–5320, 2021. 2, 3, 5

[44] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 12

[45] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019. 1

[46] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. In *ICLR*, 2020. 1, 2, 4, 7, 12

[47] Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1365–1374, 2019. 2, 3, 4, 7

[48] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 1

[49] Xinlong Wang, Tao Kong, Chunhua Shen, Yuning Jiang, and Lei Li. SOLO: Segmenting objects by locations. In *Proc. Eur. Conf. Computer Vision (ECCV)*, 2020. 5

[50] Jing Yang, Brais Martinez, Adrian Bulat, and Georgios Tzimiropoulos. Knowledge distillation via softmax regression representation learning. In *International Conference on Learning Representations*, 2020. 2, 4, 6, 7, 8

[51] Ze Yang, Shaohui Liu, Han Hu, Liwei Wang, and Stephen Lin. Reppoints: Point set representation for object detection. In *The IEEE International Conference on Computer Vision (ICCV)*, 2019. 5

[52] Zhendong Yang, Zhe Li, Xiaohu Jiang, Yuan Gong, Zehuan Yuan, Danpei Zhao, and Chun Yuan. Focal and global knowledge distillation for detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4643–4652, 2022. 2, 3, 5

[53] Zhendong Yang, Zhe Li, Mingqi Shao, Dachuan Shi, Zehuan Yuan, and Chun Yuan. Masked generative distillation. In *European Conference on Computer Vision*, pages 53–69. Springer, 2022. 1, 2, 5, 6

[54] Zhonghui You, Kun Yan, Jinmian Ye, Meng Ma, and Ping Wang. Gate decorator: Global filter pruning method for accelerating deep convolutional neural networks. *Advances in neural information processing systems*, 32, 2019. 1, 7

[55] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *Proceedings of the British Machine Vision Conference 2016, BMVC*. BMVA Press, 2016. 4, 12

[56] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR*, 2017. 2, 3

[57] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *CVPR*, pages 6848–6856. Computer Vision Foundation / IEEE Computer Society, 2018. 12

[58] Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. Deep mutual learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4320–4328, 2018. 2

[59] Borui Zhao, Quan Cui, Renjie Song, Yiyu Qiu, and Jiajun Liang. Decoupled knowledge distillation. In *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*, pages 11953–11962, 2022. 1, 2, 13

[60] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2881–2890, 2017. 6

Teacher Features-Driven Regularization for Knowledge Distillation

Supplementary Material

Training Details. We provide additional training details for the classification task. For segmentation and detection, the settings are presented in the main paper. For CIFAR-100 the initial learning rate is configured at 0.01 for MobileNet/ShuffleNet architectures and 0.05 for other architectures, and it is divided by 10 at 150th, 180th, and 210th epochs. For TinyImageNet, the learning rate is set to 0.1 for all architectures and is divided by 10 at 60th, 120th, and 160th epochs. The weight decay is set to 5×10^{-4} for all experiments. Gradient clipping with max norm equal to 10 is employed. The vanilla KD loss is included in the other methods training only if it benefits the final accuracy. The temperature parameter is set to 4. All experiments are conducted with Pytorch [35]. The computing infrastructure comprises a server equipped with 3 NVIDIA Tesla V100 GPUs with 32GB RAM with CUDA version 11.6 for classification and segmentation experiments, and a server containing 4 NVIDIA RTX A6000 GPUs with 48GB RAM with CUDA version 12.0 for detection experiments.

Implementation Details. Fig. 7 presents a detailed illustration of our method applied to a generic classification network. Typically, a model backbone is divided into 4 stages, each comprising M layers. As displayed in the diagram, stage 1 student features are first aligned to the teacher features dimensions through adaptation layer $\mathcal{A}_1$ and consequently fed to the frozen stage 2 of the teacher. The output is matched to the teacher features at the same level with our L_2 TeaFeD loss. The same procedure is repeated for all the subsequent layers, e.g. stage 2 student features are initially adapted and then given as input to teacher stage 3 to produce new features at a higher level of abstraction needed to compute our regularization term. If the backbone is partitioned into $L > 4$ stages (for example, as in MobileNet), our approach is applied to the last four stages. In this case, stage L is equivalent to stage 4 of the figure, stage L-1 corresponds to stage 3, and so forth. Note that, as regards classification architectures, the neck is a simple pooling operation.

As mentioned in the main paper, for classification models the method is employed on the (last four) backbone features. In contrast, for the segmentation task, only the last backbone feature map is considered. This feature is provided as input directly to the first head layer, given the absence of the neck. Finally, detection networks are more complex but involve an analogous implementation. In this context, four multi-level features are extracted from the neck and passed through the first layer of the detection head.

Layer	Accuracy
Conv1x1-BN	78.75±0.12
Conv3x3-BN-NL	78.60±0.10
Conv1x1-Conv3x3-Conv1x1	78.34±0.24
Conv1x1-Conv3x3(DW)-Conv1x1	78.46±0.04
Ours (Conv3x3-BN)	**78.84±0.21**

Table 8. Comparison of adaptation layers. ConvNxN indicates a convolutional layer with a NxN kernel size, while BN, NL, and DW denote batch normalization, ReLU non-linearity, and depth-wise convolution, respectively. Each convolution in the multi-layer configurations is followed by BN and NL.

Network Architectures. We evaluate performance using several teacher-student pairs involving various popular neural network architectures, including VGG [44], ResNet [18], WRN [55], ShuffleNetV1 [57], ShuffleNetV2 [32], and MobileNetV2 [39]. In these architecture labels, the number following VGG and ResNet indicates the depth of the networks. WRN-d-w refers to a Wide-ResNet with depth d and width factor w. Similar to prior studies [3, 4, 46], we adjust the number of convolution filters in intermediate layers of certain network architectures by a specified ratio R, denoted by xR in labels such as ResNet-32x4.

Adaptation Layer. We compare several implementations of the adaptation layer $\mathcal{A}$, which aligns the student features with those of the teacher. The first rows of Tab. 8 suggest that any straightforward single-layer design is beneficial for the knowledge transfer procedure, providing comparable performances. Notably, the highest accuracy is achieved with a 3x3 convolutional layer followed by a batch normalization. This configuration is employed in every experiment across all tasks. Conversely, aligning the features with a three-layer transformation mitigates the distillation effectiveness. We conjecture this is due to the additional capacity provided by the multi-layer structure. In this case, the adaptation module can learn to transform any student representation into any complex teacher representation, preventing the student model from receiving useful feedback and thus learning robust intermediate features.

Method Extension. We extend our method by modifying the design of the loss. In this setting, the features at layer ℓ are propagated to all the corresponding subsequent layers of the teacher, i.e. until the final layer L. Thus, our regularization term is computed on the output logits using Eq. 2. As

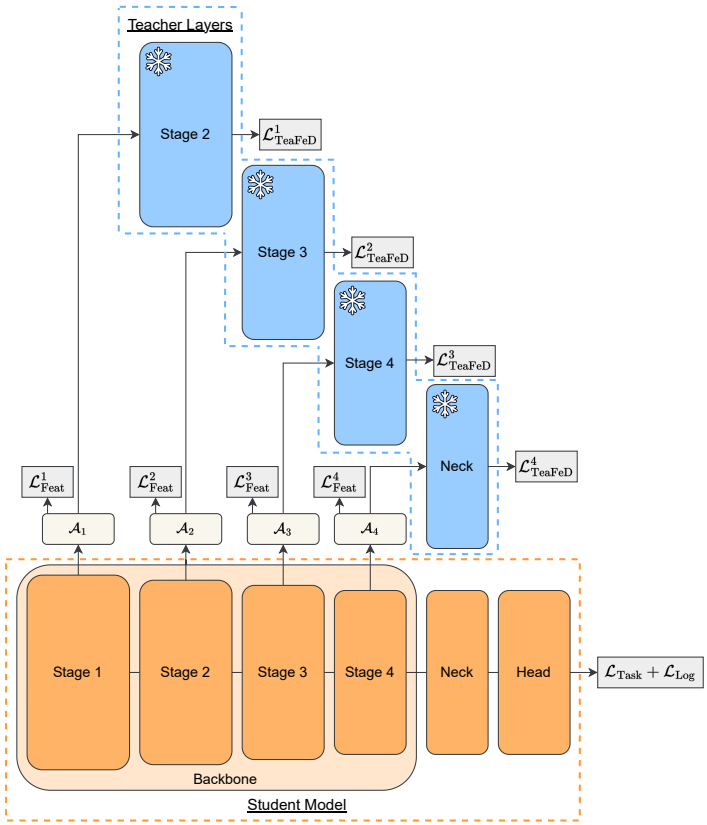


Figure 7. Detailed representation of our method implementation for a generic vision model.

presented in Tab. 9, the accuracy values from this strategy are on par with our feature-driven loss, adding no significant advantages to the distillation process. Moreover, the primary limitation of this approach is the extended training time, which increases by 30% compared to our original configuration. Additionally, in this revisited setup, we train the student with a different logit-based loss, applying DKD [59] to both $\mathcal{L}_{\text{Log}}$ and $\mathcal{L}_{\text{TeaFeD}}$. As expected, the adoption of a more effective training objective on the logits enhances the results, which can be found in Tab. 10. Despite the poor training efficiency, the setting can be leveraged as a pure logit-based distillation method (removing $\mathcal{L}_{\text{Feat}}$), boosting the student trained exclusively with $\mathcal{L}_{\text{Log}}$ when the features are not available. Furthermore, the configuration is versatile and adaptable to any logit-based technique.

More Visualizations. Fig. 8 provides additional examples of the phenomenon observed in Fig. 2 in the main paper. Despite producing intermediate features quantitatively closer to the teacher, as shown by the Euclidean and cosine distance in the upper corners, the vanilla feature matching method [38] fails to classify inputs correctly. The reason can be seen qualitatively from the activation maps computed with Grad-CAM [41], proving that our approach focuses on the same image regions as the teacher, while the baseline [38] gets distracted by irrelevant pixels with semantic or appearance patterns.

	Student	Ours	Ours$^+$	Teacher
Homog.	ResNet-116	77.54±0.13	77.65±0.16	ResNet-110x2
	VGG-8	75.01±0.06	74.81±0.05	VGG-13
	ResNet-8x4	76.64±0.16	76.63±0.17	ResNet-32x4
	WRN-40-1	75.18±0.22	75.26±0.27	WRN-40-2
Heterog.	VGG-8	75.74±0.14	75.41±0.29	ResNet-32x4
	ShuffleNetV1	76.36±0.17	76.16±0.25	VGG-13
	MobileNetV2	70.94±0.11	70.77±0.39	WRN-40-2
	ShuffleNetV2	78.84±0.21	78.43±0.07	ResNet-32x4

Table 9. Top-1 test accuracy (%) of homogeneous and heterogeneous teacher-student pairs on CIFAR-100 obtained with our extended method, denoted as +.

Student	KD	DKD	Ours$^+$ w/ KD	Ours$^+$ w/ DKD	Teacher
ShuffleNetV2 73.33	75.84±0.22	77.02±0.18	78.43±0.07	78.70±0.13	ResNet-32x4 79.21

Table 10. Top-1 test accuracy (%) on CIFAR-100 obtained with our extended method using different logit-based approaches.

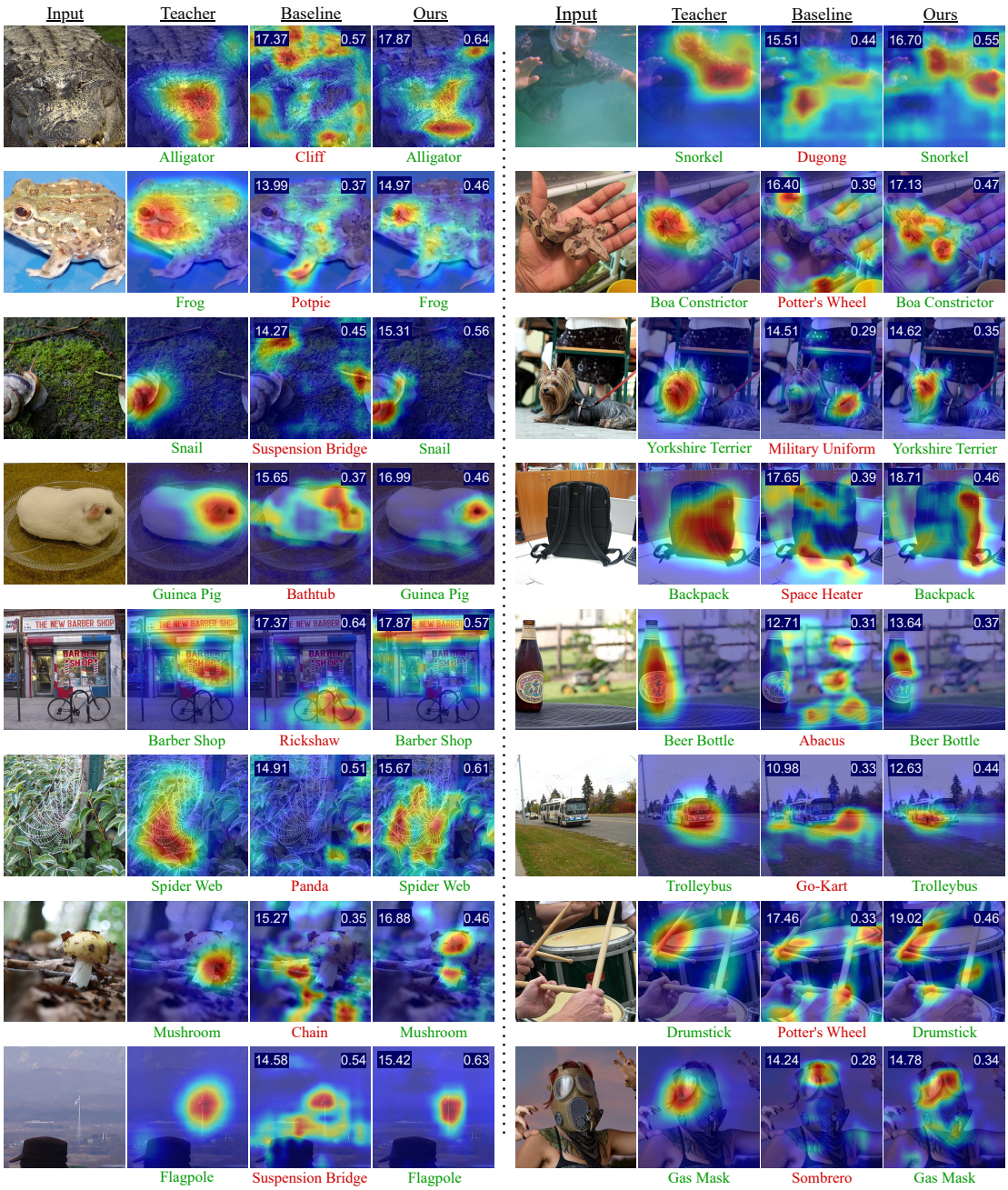


Figure 8. Visualization of the representations learned by the student with the baseline method [38] and our approach. The Euclidean and cosine distances, shown in the upper left and right corners of the students results, respectively, confirm that the baseline produces closer features to the teacher. However, it fails to classify inputs correctly as it focuses on irrelevant pixels, while our approach matches the teacher's attention. Activation maps are computed with Grad-CAM [41], while distances are computed on the last backbone feature vector.