IMF: Integrating Matched Features Using Attentive Logit in Knowledge Distillation

Jeongho Kim[1] , **Hanbeen Lee**[2] , **Simon S. Woo**[3*]
[1]Korea Advanced Institute of Science and Technology (KAIST), S. Korea
[2]NAVER Z Corporation, S. Korea
[3]Department of Artificial Intelligence, Sungkyunkwan University, S. Korea
rlawjdghek@kaist.ac.kr, gksqls5707@naverz-corp.com, swoo@g.skku.edu

Abstract

Knowledge distillation (KD) is an effective method for transferring the knowledge of a teacher model to a student model, that aims to improve the latter's performance efficiently. Although generic knowledge distillation methods such as softmax representation distillation and intermediate feature matching have demonstrated improvements with various tasks, only marginal improvements are shown in student networks due to their limited model capacity. In this work, to address the student model's limitation, we propose a novel flexible KD framework, Integrating Matched Features using Attentive Logit in Knowledge Distillation (IMF). Our approach introduces an intermediate feature distiller (IFD) to improve the overall performance of the student model by directly distilling the teacher's knowledge into branches of student models. The generated output of IFD, which is trained by the teacher model, is effectively combined by attentive logit. We use only a few blocks of the student and the trained IFD during inference, requiring an equal or less number of parameters. Through extensive experiments, we demonstrate that IMF consistently outperforms other state-of-the-art methods with a large margin over the various datasets in different tasks without extra computation.

1 Introduction

Although deep learning methods have been shown to achieve great performance in various tasks over computer vision and natural language processing [Dosovitskiy *et al.*, 2020; Brown *et al.*, 2020; Ramesh *et al.*, 2021], they generally require large datasets, model parameters, and extensive computations. Therefore, deploying complex models on mobile devices is one of the major challenges due to the device's limited computational capabilities. Thus, many researchers have proposed approaches to overcome such limitations. In particular, knowledge distillation (KD) [Hinton *et al.*, 2015] has been proposed to effectively transfer the knowledge of a large model ("teacher") to a smaller model ("student").

The existing KD methods can be categorized into one of two groups: 1) softmax representation distillation [Hinton *et al.*, 2015; Park *et al.*, 2019; Passalis and Tefas, 2018; Tian *et al.*, 2019; Yang *et al.*, 2021; Lee *et al.*, 2022] and 2) latent (intermediate) feature matching [Romero *et al.*, 2014; Zagoruyko and Komodakis, 2016; Yim *et al.*, 2017; Tung and Mori, 2019]. The first approaches have explored how to generate and transfer better final representations of the model by minimizing the difference of the softmax representation between the teacher and student model using KL divergence proposed by Hinton *et al* [Hinton *et al.*, 2015]. On the other hand, the second approach has focused on improving latent feature map matching by mimicking the intermediate representation that works as a strong indicator to help learn the terminal representation [Gou *et al.*, 2021].

Most KD approaches have focused on enhancing the performance of the student model without changing its underlying structure. However, such approaches inevitably have to select the matching student model that satisfies the accuracy, inference time, and a number of model parameter requirements. To reduce the inference time, one can choose the small models for convenience. However, due to the unsatisfactory performance from severe capacity gap or model structures [Cho and Hariharan, 2019], it is needed to re-select and retrain the student, which can be inefficient and cumbersome. Moreover, not much research has been conducted in the area of KD to explore the performance vs. computational cost trade-off, focusing on the flexible and adaptive student model architecture. To alleviate the issues above, we introduce the flexible and adaptive KD architecture to improve performance while maintaining efficiency.

In this paper, we propose Integrating Matched Features using Attentive Logit in Knowledge Distillation (IMF), to improve the student performance by combining the outputs from each intermediate feature distiller (IFD) that the teacher model trains. Moreover, IMF is designed to be an adaptive architecture so that the student model can be flexibly configured to improve the performance, while lowering the complexity via the different number of distilled IFDs in branch networks. In particular, each IFD takes the student's intermediate feature map as input and produces its latent representation by exploiting the element-wise normalized attentive (ENA) layer to learn and mimic the teacher's knowledge better for the higher performance, achieving higher perfor-

[*] Corresponding author

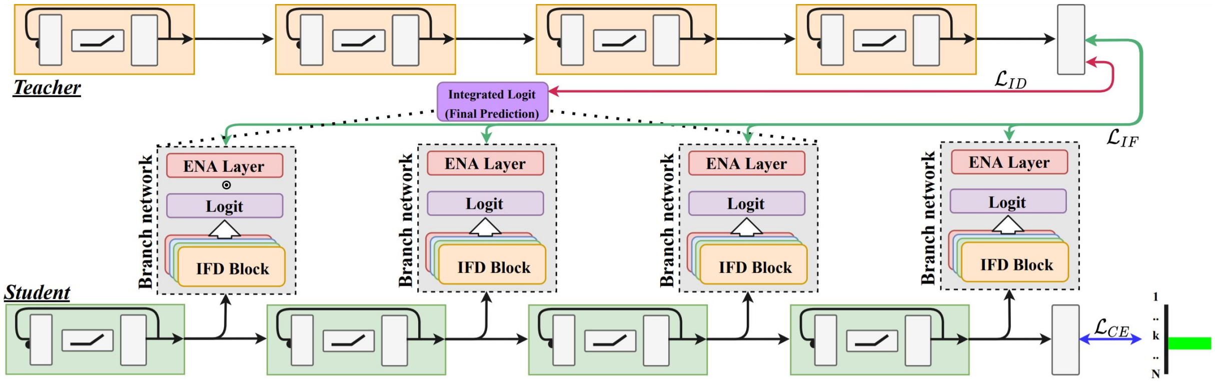

Figure 1: Overview of our proposed integrating matched feature framework (IMF). For N blocks (N=4), intermediate feature distiller (IFD) is shown as stacked blocks in the middle, and element-wise normalized attentive (ENA) layer is shown in a red block followed by IFD. In the inference phase, we use integrated logit from the 3 IFDs as our final prediction instead of forwarding the existing fully connected layer.

mance. Moreover, combined outputs of IFD are also trained to mimic the teacher to boost the performance further. Hence, with the smaller final student's model capacity compared to the original student model, IMF can achieve higher performance. Through extensive experiments, we demonstrate that our approach outperforms the best performing state-of-the-art baselines, and shows robustness in distilling knowledge from various teacher models to the student models.

2 Related Work

2.1 Knowledge Distillation

Hinton *et al.* [Hinton *et al.*, 2015] first proposed KD to boost the performance of a student network by transferring the knowledge from a teacher network, showing that the student can learn not only the hard-label information but also the soft-label information from the softmax output of the teacher. On the other hand, feature map-based approaches [Romero *et al.*, 2014; Zagoruyko and Komodakis, 2016] are proposed, to reduce the distance between the respective feature maps of a teacher and student. In terms of direct matching features between teacher and student network, relational knowledge distillation (RKD) [Park *et al.*, 2019] considered the input examples as mutual relations and transferred these relations instead of distilling softmax representation. Furthermore, contrastive representation distillation (CRD) was proposed by Tian *et al.* [Tian *et al.*, 2019], which utilizes contrastive learning for the student to capture distance-based information (data point) from the teacher's representation with an additional layer during training; however, in a regression task, CRD cannot be easily applied because positive and negative pairs are difficult to define and obtain.

Recently, Yang *et al.* [Yang *et al.*, 2021] devised a novel distillation via softmax regression representation learning (SR) to enable direct feature matching at the penultimate layers of the teacher and student. First, they decoupled the latent feature and the softmax classifier. Then, for successful representation learning, they employed a frozen teacher classifier to train the student's penultimate layer feature in the teacher's classifier, and they achieved state-of-the-art performance. Also, weight soft label distillation (WSL) was proposed by Zhou *et al.* [Zhou *et al.*, 2021] to apply a bias-variance trade-off to the KD method. They observed that the bias-variance trade-off varies sample-wisely during training with soft label information, which affects the negative results in KD. Especially, they filtered out larger trade-off samples and achieved higher performance. Our approach greatly differs from the above methods as we effectively combine intermediate features with attentive logits from features of student's blocks and inference only with those blocks.

2.2 Branch Network

Before the advent of skip connections [He *et al.*, 2016], Inception [Szegedy *et al.*, 2015] was the state-of-the-art method to achieve the best performance on ImageNet [Deng *et al.*, 2009] by using auxiliary classifiers. In particular, an auxiliary classifier can encourage discrimination in class labels in the shallow layers and effectively provide regularization to intermediate features, directly injecting cross entropy loss to the intermediate classifiers. Not only are branch networks used to enhance classification performance, but also Teerapittayanon *et al.* proposed using branch networks for an early exit in the inference process [Teerapittayanon *et al.*, 2016]. Based on an early layer of a CNN network which is sufficient for classifying many data points, they achieved compressed inference time with comparable performance in several CNN models.

Self-distillation [Zhang *et al.*, 2019], one of the newly proposed KD concepts as mentioned, enhanced the performance of DNNs without the supervision of the teacher by transferring the knowledge of deep layers to shallow layers via branch networks; however, each branch network, which is much smaller than teacher models, cannot provide refined information because of its limited student's capacity [Ji *et al.*, 2021]. Therefore, its performance enhancement is much less compared to the KD methods using teacher networks. The early exit and aforementioned methods used branch networks for performance improvement and inference time reduction, where we similarly adapt branch networks as feature distillers.

However, in our methods, we first propose a novel framework, in which the branch networks densely distill the logit of the teacher to the branch networks, and takes the intermediate multi-scale feature maps of the student network as inputs. Moreover, the branch networks are used as predictors without extra computation, while discarding the existing classifier and the last MLP block in the inference phase. By changing the forwarding path, the IMF flexibly controls the model size to have the same or even less number of parameters than that of the original student networks, while achieving higher performance. Exploiting several branch networks, we propose a novel ENA layer that provides adjustable gradients to each branch network from ID loss. We compare our approach to all aforementioned baseline approaches for our experiments – KD, RKD, CRD, SR, PKT, FN, AT, and WSL.

3 Our Approach

Intermediate Feature Distiller (IFD). We first denote t_i and s_i as the ith layer's feature map of the teacher and student network, respectively. And, z^t and z^s represent the teacher's and student's logit, respectively as well. Then, the softmax function is defined as follows: $\sigma(z_i) = \frac{exp(z_i/\tau)}{\Sigma_j(exp(z_j/\tau))}$, $i = 1, ..., M$, where z_i is the ith class logit in M classes, and τ is the temperature. The IFD is trained with the teacher's knowledge (softmax representation), forwarded by each student's intermediate layers as inputs. Next, we designed several branch networks to sufficiently classify the input instances and utilized the trained IFD block as an auxiliary network to improve the overall performance, as shown in Fig. 1.

More formally, we define F^i_{IFD} as the ith IFD network to produce the output from s_i, and s'_i as $F^i_{IFD}(s_i)$ for simplicity. Next, we define S^i_{IF} as the softmax output of ith IFD in Eq 1, as follows:

$$\mathcal{S}^i_{IF} = \sigma\left(F^i_{IFD}(s_i)\right) = \sigma(s'_i). \quad (1)$$

Then, we first design a naïve version of intermediate feature matching loss $\mathcal{L}_{IF}$ to demonstrate our training concept. Here, each IFD is individually trained, taking into account the classification object. Specifically, in $\mathcal{L}_{IF}$, we consider not only distillation loss from the teacher but also cross entropy loss for hard label information. Finally, we formulate the intermediate feature matching loss as follows:

$$\mathcal{L}_{IF} = \sum_{i=1}^{N_{block}} \tau^2 \cdot (1-\alpha) \cdot D_{KL}(\sigma(s'_i) \mid\mid \sigma(z^t)) \quad (2)$$
$$+ \alpha \cdot \mathcal{L}_{CE}(s'_i, y)],$$

where N_{block} is the number of blocks, and α is the weight variable between D_{KL} and L_{CE}, and we empirically set α = 0.2.

The intuition behind our design of $\mathcal{L}_{IF}$ is that the teacher provides a *proper* supervision at each stage of a student. Generally, shallower latent features of the student tend to result in weaker representational power of the student. These features are much different from the teacher representation, making training the student model difficult. On the other hand, our framework uses a branch network by training IFDs, where shallow layers and IFDs can easily learn the teacher representation without the risk of losing knowledge.

Element-wise Normalized Attentive (ENA) Layer. In addition, we add the element-wise normalized attentive (ENA) layer to the last layer of each IFD block. We designed the network such that each layer in the network can attend to the representation by itself. Therefore, each IFD can emphasize its class logits by attaching the attentive layer with element-wise multiplication.

As an example, Fig. 2 illustrates that the final results can be changed by ENA layers. Specifically, in Fig. 2, let us assume that the red bars represent the correct labels (e.g., Panda) and the blue bars represent incorrect labels (e.g., Gibbon). Simple aggregation from 3 IFD logits would yield an incorrect result higher than that of the red bars (as shown at the bottom). However, multiplying class-wise weight parameters from the ENA layer provides more weight to the confident class (Panda) in the first IFD block. Therefore, the IFD layer combined with the ENA layer presents optimal inferences for individual classes according to the depth of the network.

Furthermore, the ENA requires only negligible additional parameters and computational cost into an IFD during training and inference time. For example, in CIFAR-100 [Krizhevsky and Hinton, 2009], which has 100 classes, 100 parameters are added to the output logits of each IFD layer. Therefore, using four blocks, as shown in Fig. 1, we can achieve performance enhancement with only additional 400 parameters in total. Lastly, to prevent gradient exploding in the early training phase, we apply z-score normalization to each ENA layer at every training iteration. By introducing new attentive layers, we can further refine Eq. 2 to the following equation:

$$\mathcal{L}_{IF} = \sum_{i=1}^{N_{block}} [\tau^2 \cdot (1-\alpha) \cdot D_{KL}(\sigma(v^i_{ENA} \odot s'_i) \mid\mid \sigma(z^t)) \quad (3)$$
$$+ \ \alpha \cdot \mathcal{L}_{CE}(v^i_{ENA} \odot s'_i, \ y)],$$

where $\odot$ is hadamard product and v_{ENA} is a vector for emphasizing their softmax representation.

Integrated Distillation (ID) Loss. For integrating each well-trained IFD block, we introduce integrating distillation (ID) loss $\mathcal{L}_{ID}$. Motivated by the early exit method that illustrates that easy tasks can be effectively classified in early layers with a relatively small student capacity, the method can provide fast inference time. Therefore, we focus on exploiting early layers to improve classification performance and inference time rather than exploiting the fixed student model commonly used in traditional KD methods. Since attaching the ENA layers boosts the confidence of each IFD block to an instance, $\mathcal{L}_{ID}$ is a key component of combining the results from the separated branch networks. With the ENA layer, ID loss can distill and distribute the refined information from the teacher model to the IFD block through the integrated logit from several small branch networks. Therefore, we define

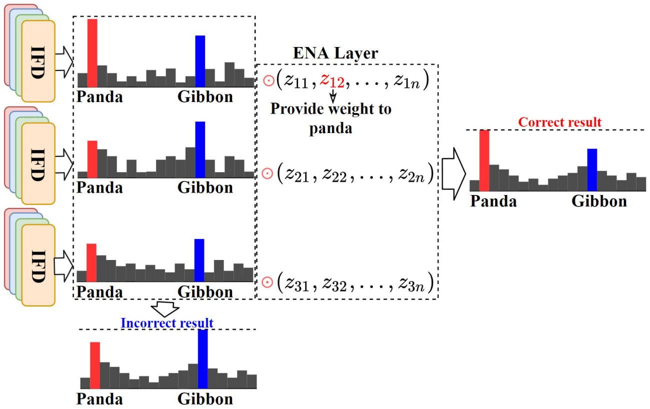

Figure 2: Element-wise multiplication is applied for each class logit in IFD blocks. Since each block learns the representation of input images differently, each adaptive weight parameters provide weight to the prediction. A correct label (e.g., Panda) in the red bar from the first IFD shows the highest confidence among the three blocks and takes attention from the first ENA layer. Hence, the correct final result can be achieved. On the other hand, the simple aggregation would yield the incorrect result (e.g., Gibbon).

$\mathcal{L}_{ID}$ as follows:

$$\mathcal{L}_{ID} = D_{KL}\left(\sigma\left(\sum_{i\in\mathcal{D}} v^i_{ENA} \odot s'_i\right) \Bigg|\Bigg| \sigma(z^t)\right), \tag{4}$$

where $\mathcal{D}$ is a set of IFD's index for integrated logit as a final prediction, as shown in Fig. 1. In $\mathcal{D}$, it is not compulsory to include all IFDs that are trained with $\mathcal{L}_{IF}$.

Finally, we can construct our overall optimization objective as follows:

$$\mathcal{L} = \mathcal{L}_{CE} + \alpha \cdot \mathcal{L}_{IF} + \beta \cdot \mathcal{L}_{ID}, \tag{5}$$

where α and β are the weight variable set as 10 and 30, respectively, and $\mathcal{L}_{CE}$ is a cross entropy loss between z^s and class label.

Inference. Traditional KD methods do not alter the structure of the student network. In contrast, IMF is not restricted to the architecture of the student network, as the architecture of the student network can be adjusted to achieve higher performance. In the inference stage, IMF does not use the entire student network but uses only a part of the network. As illustrated in Fig. 1, a partial network of the student is integrated with $\mathcal{D}$ to become an inference model to produce the final outputs. We can control the number of IFD layers of $\mathcal{D}$ to make the model lighter or to produce the highest performance. In our experiment, we can empirically obtain the best performance when we use $N_{block} - 1$ IFD layers in inference. Therefore, we can adjust the number of IFDs not to exceed the computational cost of the student.

Theoretical Support. We provide the following theoretical results for characterizing the interaction and validity of $\mathcal{L}_{IF}$, and the further need for $\mathcal{L}_{ID}$ and the ENA layer to achieve higher performance from the gradient perspective.

First, we can obtain $\frac{\partial \mathcal{L}_{ID}}{\partial s'_i}$ as follows:

$$\begin{aligned}
\frac{\partial \mathcal{L}_{ID}}{\partial s'_i} &= \frac{\partial}{\partial s'_i}\left[D_{KL}\left(\sigma\left(\sum_{i\in\mathcal{D}} v^i_{ENA} \odot s'_i\right) ||\sigma(z^t)\right)\right] \\
&= \frac{\partial}{\partial s'_i}\left[-\sigma(z^t)log\left(\sigma\left(\sum_{i\in\mathcal{D}} v^i_{ENA} \odot s'_i\right)\right)\right] \\
&= v^i_{ENA} \odot \left(-\sigma(z^t) + \sigma\left(\sum_{i\in\mathcal{D}} v^i_{ENA} \odot s'_i\right)\right).
\end{aligned} \tag{6}$$

Hence, the gradient from $\mathcal{L}_{ID}$ depends on each ENA layer and all outputs of branch networks used in inference. Since $\mathcal{L}_{ID}$ is necessary for $\mathcal{L}_{IF}$, all IFD outputs $f^i(s'_i)$ would be matched to the value between the teacher's logit and hard label if it is theoretically optimized.

However, due to the limited capacity and different path lengths of IFD (student) blocks, this solution cannot be achievable. Therefore, we introduce $\mathcal{L}_{ID}$ and the ENA layer (v_{ENA}) in IMF, where, in the Eq. 6, the gradient from $\mathcal{L}_{ID}$ is affected by ENA layer and all output of the IFD blocks $\left(\sum_{i\in\mathcal{D}} v^i_{ENA} \odot s'_i\right)$. It enables the proposed ENA layer, and $\mathcal{L}_{ID}$ to be *attentive* to each value of the branch networks.

4 Experimental Results

Training details. Backbone architecture and training settings for experiments are similar to the recent research [Tian *et al.*, 2019]. In our method, we conduct a grid search to choose the α and β values in Eq. 5 from $\{10, 20, 30, 40\}$. The IFD block has the same structure in all experiments and model architectures. Specifically, we used a block structure of DepthConv(3×3) - PointConv(1×1) - BN & ReLU. As the layers became deeper and the resolution decreased, we incrementally reduced the number of blocks by one. Only the channel size of the conv layer was adjusted, based on the size of the original student network. Also, for a fair comparison, we repeated each experiment three times and calculated the mean of the scores.

Comparison. Since IMF uses the flexible architecture with the intermediate IFDs, it is not easy to directly compare IMF with other baselines with the fixed student structures. Instead, to provide a fair comparison as much as possible during inference, we keep our entire computational cost to be the same as the total computation cost of the compared student models in baselines. In other words, the final student model in IMF would not be equivalent to the original student model because IMF-trained student model has a structural difference compared with its original model. However, we can clearly compare the performance of each approach by fixing the total computational cost.

4.1 Performance Evaluation

CIFAR-100. We compared top-1 accuracy for 13 teacher-student pairs. Furthermore, we calculated the number of parameters of a student model to verify that our method improves performance from a trade-off between the model's

Teacher **Student**	**VGG19** **VGG8**	**VGG19** **VGG11**	**VGG13** **VGG8**	**ResNet56** **ResNet20**	**ResNet110** **ResNet20**	**ResNet110** **ResNet32**	**ResNet32x4** **ResNet8x4**	**WRN-40-2** **WRN-16-2**	**WRN-40-4** **WRN-16-4**
Teacher	74.71	74.71	74.64	73.92	74.08	74.08	79.2	76.32	79.67
T.Params Ratio	(*100%*)	(*100%*)	(*100%*)	(*100%*)	(*100%*)	(*100%*)	(*100%*)	(*100%*)	(*100%*)
Student	70.67	71.78	70.67	69.42	69.42	71.24	73.51	73.41	77.48
S.Params Ratio	(*19.7%*)	(*46.1%*)	(*41.9%*)	(*32.3%*)	(*16.0%*)	(*27.2%*)	(*16.5%*)	(*31.1%*)	(*30.8%*)
KD [NeurIPS'14]	72.42	74.15	73.4	70.97	70.72	73.58	73.97	74.88	77.46
AT [ICLR'17]	72.27	73.89	73.32	70.67	70.73	73.41	74.88	74.91	77.60
FN [ICLR'15]	72.33	74.28	73.35	71.02	70.81	73.34	73.89	75.21	77.51
RKD [CVPR'19]	72.73	74.02	73.69	71.08	70.71	73.28	73.71	74.82	77.74
CRD [ICLR'20]	73.04	73.92	74.31	71.80	71.69	74.24	76.05	75.86	75.59
WSL [ICLR'21]	70.49	71.73	71.52	72.01(72.15)	71.14(72.19)	72.91(74.12)	73.81(76.05)	74.75	76.99
SR [ICLR'21]	73.68	74.21	73.31	70.86	70.78	73.1	75.71	75.49	79.05 (79.58)
Ours	**76.14**	**77.14**	**77.09**	**73.69**	**73.92**	**75.77**	**78.42**	**77.37**	**79.95**
O.Params Ratio	(*19.2%*)	(*45.8%*)	(*40.9%*)	(*32.1%*)	(*15.9%*)	(*27.1%*)	(*15.5%*)	(*30.9%*)	(*30.8%*)

Table 1: Avg. top-1 accuracies (%) with CIFAR-100 using the same architecture style between teacher and student. The best results are highlighted in bold, and the higher performance than the teacher is underlined. Also, we denote the ratio of the parameter numbers of a student and our ensemble model to the teacher in the parenthesis. We reran and reproduced the results of all methods, and we report the mean of the three trials. In the case of WSL and SR, we report the scores from their respective papers in the parenthesis if the same teacher and student pair with ours is experimented in the reference paper because reimplemented accuracies are not as high as the original.

Teacher **Student**	**RN110** **SN-V1**	**RR32x4** **SN-V1**	**WRN-40-6** **SN-V2**	**WRN-40-2** **N-V2**
Teacher	74.08	79.77	80.75	76.32
T.Params Ratio	(*100%*)	(*100%*)	(*100%*)	(*100%*)
Student	71.23	71.23	73.35	73.35
S.Params Ratio	(*54.66%*)	(*12.77%*)	(*6.73%*)	(*60.11%*)
KD [NIPSW'14]	75.94	74.64	74.51	76.29
AT [ICLR'17]	76.49	74.67	74.57	76.74
FN [ICLR'15]	76.25	74.76	74.64	76.56
RKD [CVPR'19]	76.24	74.44	74.65	76.22
CRD [ICLR'20]	76.44	75.53	76.33	77.27
WSL [ICLR'21]	73.91	70.7 (75.46)	72.32	74.43
SR [ICLR'21]	75.15	74.18 (75.66)	72.83	74.87
Ours	**77.23**	**76.55**	**78.07**	**78.6**
O.Params Ratio	(*51.2%*)	(*12.3%*)	(*6.1%*)	(*53.1%*)

Table 2: Avg. top-1 accuracies (%) with CIFAR-100 using different architecture styles between the teacher and student. The best results are highlighted in bold, and the higher performance than the teacher is underlined. For simplicity, we abbreviate ResNet and ShuffleNet as RN and SN, respectively.

size and performance. As shown in Table 1, our IMF outperforms all other methods across all backbone architecture pairs. Especially, our IMF is the only model that outperforms the teacher model in VGG, ResNet110, and WRN.

In addition, our method has fewer parameters than the original student model since we do not utilize an entire structure of the student model in the inference stage. For example, in the ResNet32x4-ResNet8x4 pair, our IMF method has 15.5% of the teacher's parameters but achieves 4.91% performance improvement compared to the original student model. Moreover, as shown in Table 2, IMF also consistently outperforms other KD methods in different teacher-student model architectures.

CIFAR-10. To demonstrate that IMF achieves performance gain in other image classification tasks, we experiment with CIFAR-10. As shown in Table 3, our IMF obtains the performance improvement of up to 1.3% compared to all other competing methods, even though all results are above 90%. Moreover, IMF shows better performance than the teacher networks in VGG19-VGG8 and VGG19-VGG11 pairs, as same with the CIFAR-100 experimental results. Therefore, in the more simple dataset, our IMF outperforms other competing works.

Teacher **Student**	**VGG19** **VGG8**	**VGG19** **VGG11**	**VGG13** **VGG8**	**RN56** **RN20**	**RN110** **RN20**	**RN32x4** **RN8x4**
Teacher	93.86	93.86	94.36	93.94	94.45	95.52
Student	91.86	92.51	91.86	92.63	92.63	92.81
KD [NIPSW'14]	92.78	92.90	93.06	93.13	93.13	93.86
AT [ICLR'17]	92.97	92.95	93.09	93.19	93.28	93.72
FN [ICLR'15]	92.83	93.03	93.16	93.19	92.94	93.87
RKD [CVPR'19]	92.77	92.91	93.07	93.25	93.16	93.92
CRD [ICLR'20]	92.97	93.03	93.02	93.23	93.02	94.01
SR [ICLR'21]	92.42	93.2	92.8	93.13	93.24	94.08
Ours	**94.05**	**94.15**	**94.05**	**93.65**	**93.69**	**95.08**

Table 3: Avg. top-1 accuracies (%) with CIFAR-10 in the same architecture between teacher and student models. The best results are highlighted in bold, and the scores are underlined in the case of outperforming the teacher's performance. For simplicity, we abbreviate ResNet as RN.

ImageNet. To evaluate IMF performance in a large-scale dataset, we demonstrated experimental results on the ImageNet dataset. In this experiment, we selected the ResNet family as the same base architecture style. As shown in Table 4, IMF outperforms baseline methods with a fewer number of parameters. In detail, IMF's top-1 performance increases by 2.63% and 0.94%, compared to the baseline student as well as the state-of-the-art, SR, respectively. Similar to CIFAR-100, the number of parameters in our model during the inference phase is less than that of the student model, as shown in Table 7.

Facial Keypoints Detection. We conducted experiments not only on the popular classification datasets but also on the

	RN34	RN18	KD	AT	RKD	CRD	SR	Ours
t1	73.71	70.04	70.68	70.59	71.34	71.17	71.73	**72.67**
t5	91.42	89.48	90.16	89.73	90.62	90.13	90.60	**91.10**

	Teacher	Student	Ours
# Param. **(Ratio%)**	21.80M (100%)	11.69M (53.6%)	11.66M (53.5%)
# FLOPs **(Ratio%)**	367M (100%)	182M (49.5%)	178M (48.5%)

Table 4: Top-1 (t1) and top-5 (t5) accuracies (%) on ImageNet in ResNet34 (RN34) as a teacher and ResNet18 (RN18) as a student. Moreover, we denote the number of parameters (Param.) and FLOPs with a ratio of students and our method to the teacher for a fair comparison. Also, the best results are highlighted in bold.

regression dataset. Our method is compared with KD, AT, PKT, FN, and RKD with one same and two different network architecture styles, as shown in Table 6. We achieve higher performance than all the other baselines in the different architecture styles, except for the VGG19-VGG8 pair, where our method shows the second-highest performance. However, the overall performance of IMF is much better than others.

4.2 Attributions of IMF

For ablation, we explore the effectiveness of each proposed loss and ENA layer through an ablation study on CIFAR-100.

Effect of $\mathcal{L}_{ID}$. First, we started the experiment using only IFD, as shown in Config. A in Table 5. By only using $\mathcal{L}_{IF}$, accuracies significantly increased compared to that of the original student model. Additionally, we conducted Config. B experiment by adding $\mathcal{L}_{ID}$ to Config. A to integrate the IFD blocks. Interestingly, we observed that $\mathcal{L}_{ID}$ is helpful in increasing the performances in relatively small student networks (i.e., ResNet20, ResNet32, WRN-16-2, and ShuffleNetV2). Therefore, we found that these models do not have sufficient capacity to learn the representations to classify the images. To compensate for this weakness, $\mathcal{L}_{ID}$ integrates the small branch networks so that the capacity of the merged model is much superior to the capacity of individual models.

Effect of ENA Layer. To demonstrate the effectiveness of weight to the confident classes from several IFD blocks, we explore the effect of the ENA layer. As mentioned above, we define Config B. as a combination of Config. A and $\mathcal{L}_{ID}$. This shows impressive results that the accuracies in ResNet pairs increase, compared with the Config. A. As shown in the second and third rows in Table 5, all performances are much higher in Config. C, which attaches the ENA layer to Config. B, except in the VGG19-VGG8 pair. Therefore, as demonstrated in Section 3, ENA layer provides a weighted gradient to each branch network from ID loss, which shows the best-generalized performance over the teacher-student model pairs.

Effect of $\mathcal{L}_{IF}$. Lastly, we evaluate the performance in Config. D of Table 5, which removes $\mathcal{L}_{IF}$ from Config. C. We can observe that this combination still outperforms students' performance. However, compared with Config. C, it shows performance degradation in all pairs except for ResNet110-ResNet32. Interestingly, compared with ResNet20 and ResNet32 student networks, the accuracies decreased greater in VGG networks (more than 3%), which have much more parameters than ResNet20 and ResNet32, demonstrating the benefits of $\mathcal{L}_{IF}$.

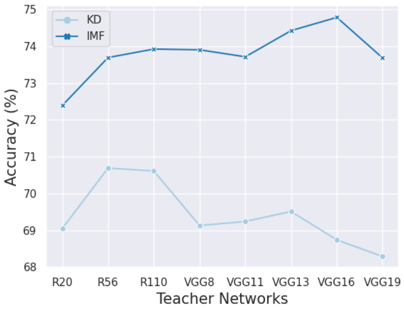

Figure 3: Top-1 accuracies (%) on CIFAR-100 in ResNet20 as a student and various teachers (i.e. ResNet and VGG family). Only distilling knowledge with KD [Hinton *et al.*, 2015] shows performance degradation in more powerful teacher networks (e.g., VGG16 and 19). Meanwhile, IMF shows robust performance in overall teacher networks. For simplicity, we abbreviate ResNet as R.

4.3 Robustness and Generalization to Various Teacher Models

The previous work [Cho and Hariharan, 2019] shows that the knowledge distillation may not be effective when the student's capacity is too low to successfully mimic the teacher, even the teacher shows better performance. Therefore, finding an adequate teacher and student network pair considering model architecture and capacity is a critical research problem. In this section, we conduct extensive experiments to explore whether our proposed method is robust and generalizable to different teacher-student architecture pairs with varying model capacity. We chose ResNet20 as a student model, while using three ResNet and five VGG families with various sizes as teacher models.

As shown in Fig. 3, ResNet20 with KD [Hinton *et al.*, 2015] shows the best performance with additional supervision from ResNet56 as a teacher. On the other hand, it has a slight performance decrease in the ResNet110 as a teacher, even the latter teacher model is more powerful, as shown by the work [Cho and Hariharan, 2019]. Moreover, in the VGG family as teacher networks, there are marginal performance improvements in the three smallest models, VGG8, VGG11, and VGG13, but we observe a gradual decrease in the performance of the teacher network as the capacity increases.

On the other hand, interestingly, our proposed method achieves performance improvement not only in ResNet but also in VGG as a teacher, even though the teachers have the different architectures and large model capacity. Specifically, compared to the original KD, which shows the second-worst performance in ResNet20-VGG16 pair, IMF shows the highest accuracy, which is higher than the performance of ResNet20 reported in the Table 1. Moreover, compared to KD [Hinton *et al.*, 2015], IMF shows lower variance but higher mean accuracy. Therefore, our proposed IMF is ro-

Teacher **Student**	**VGG19** **VGG8**	**VGG19** **VGG11**	**VGG13** **VGG8**	**RN56** **RN20**	**RN110** **RN20**	**RN110** **RN32**	**RN32x4** **RN8x4**	**WRN-40-2** **WRN-16-2**	**WRN-40-4** **WRN-16-4**	**WRN-40-6** **ShuffleNetV2**
Student	70.67	71.78	70.67	69.42	69.42	71.24	73.51	73.41	77.48	73.35
$A.\ \mathcal{L}_{IF}$	75.88	76.84	76.66	72.63	72.83	74.86	78.02	76.28	79.55	77.02
$B.\ \mathcal{L}_{IF}$ + $\mathcal{L}_{ID}$	**76.21**	76.56	76.61	73.35	73.65	75.46	77.78	76.96	79.68	77.78
$C.\ \mathcal{L}_{IF}$ + $\mathcal{L}_{ID}$ + ENA Layer (Ours)	76.14	**77.14**	**77.09**	**73.69**	**73.92**	75.77	**78.42**	**77.37**	**79.95**	**78.07**
$D.\ \mathcal{L}_{ID}$ + ENA Layer	72.76	73.17	74.9	73.27	73.88	**75.98**	77.03	75.22	78.94	75.87

Table 5: Avg. top-1 accuracies (%) as ablation study for our approaches. Starting from the primary student network, we add our proposed IFD block with ENA layer and ID loss. The best performances are highlighted in bold. For simplicity, we abbreviate ResNet as RN.

Teacher **Student**	**VGG19** **VGG8**	**ResNet32x4** **ShuffleNetV2**	**WRN-40-2** **ShuffleNetV2**
Teacher	3.002	3.169	3.354
Student	4.637	5.272	5.272
KD	4.425	4.840	5.194
AT	4.368	4.503	5.038
PKT	4.559	4.514	4.755
FN	4.319	4.883	5.608
RKD	**4.082**	3.987	3.925
Ours	4.203	**3.632**	**3.734**

Table 6: $L1$ distance with WFLW dataset in same architecture and different architecture between the teacher and student.

	RN8x4					**RN32x4**
	Block$_{1}$	**Block$_{1:2}$**	**Block$_{1:3}$**	**Block$_{1:4}$**	**+MLP**	
Param	0.9K	59K	0.29M	1.21M	**1.23M**	7.43M
Ratio%	*0.01%*	*0.79%*	*3.88%*	*16.24%*	***16.59%***	*100%*
FLOPs	0.95M	60.06M	118.98M	177.80M	**177.84M**	1085.6M
Ratio%	*0.10%*	*5.55%*	*10.96%*	*16.38%*	***16.38%***	*100%*

	IMF				**RN32x4**
	IFD$_{1}$	**IFD$_{1:2}$**	**IFD$_{1:3}$**	**IFD$_{1:4}$**	
Param	0.25M	0.68M	**1.22M**	2.34M	7.43M
Ratio%	*3.33%*	*9.21%*	***16.53%***	*31.46%*	*100%*
FLOPs	15.64M	101.68M	**171.21M**	232.69M	1085.6M
Ratio%	*1.44%*	*9.37%*	***15.77%***	*21.43%*	*100%*

Table 7: Ablation study for the number of parameters and FLOPs. All the rows of parameters and FLOPs are cumulative results. All the percentages on the parameters and FLOPs are based on that of the teacher model. Note that we only use the three IFD blocks in ResNet32x4-ResNet8x4 (RN32x4 - RN8x4) structure so that our final parameters and FLOPs in inference are sums of the three IFD and residual blocks of the RN8x4, which are highlighted in bold.

bust and generalizable to different model pairs and capacities, which can reduce the computational time and cost to find the best teacher and student pair.

4.4 Parameters and FLOPs

Our method is flexible in transforming the fixed model architecture into smaller student networks. While our final model can be even smaller than the original student network, our final model was able to achieve higher performance. In particular, we adapted our method to the student model to obtain approximately equal or fewer parameters as well as FLoating point Operations (FLOPs). In this section, we calculated the FLOPs and model parameters of IMF in detail.

We use ResNet32x4-ResNet8x4 pair as teacher and student on CIFAR-100. In this case, we use the three IFD blocks for integrated logit. In Table 7, we reported the parameters and FLOPs of the student model and our IMF. As shown in Table 7, we can observe that Block$_{1:3}$ has much fewer parameters than that of Block$_{1:4}$. That is, if we early exit on the third block, we can add branch networks (i.e., IFD$_1$, IFD$_2$, and IFD$_3$) to the student model instead of forwarding the fourth block and MLP layer in the inference phase. Accordingly, it is notable to observe that the final parameters and FLOPs of IMF can be approximately similar to or less than that of student models.

	Student	**Mean (%)**	**Variance**
KD [NeurIPS'14]	ResNet20	69.46	0.72
Ours	ResNet20	**73.83**	**0.44**

Table 8: Mean and variance values of student accuracies over 8 different teacher networks. Compared with existing knowledge distillation, our method shows higher accuracy but stable.

5 Limitation and Future Work

In this work, we focused on image classification and regression tasks, and presented the theoretical analysis to characterize the validity of our proposed methods. More research would be needed to determine the number of branch networks to find the optimal numbers for the depth of the original backbone network and IFD blocks as well as the branch networks. For future work, we plan to incorporate additional KD methods and validate our findings.

6 Conclusion

We proposed a novel flexible knowledge distillation framework, IMF, integrating matched features using attentive logit. In IMF, the student is directly trained by teacher information by utilizing an intermediate feature distiller (IFD) in a branch network. During the inference time, we use the combined logits from learned IFDs instead of the entire student model. We conducted extensive experiments with image classification as well as facial keypoint detection tasks over 4 datasets. In image classification, our method significantly outperforms 7 baselines in 10 same and 4 different backbone architecture styles with an equal or less number of parameters. Furthermore, we also demonstrate that our method is effective in keypoint detection, compared with 5 baselines. IMF not only achieved a significant performance gain regarding classification tasks as well as a regression task, without requiring an extra computational cost.

Acknowledgments

This work was partly supported by Institute for Information & communication Technology Planning & evaluation (IITP) grants funded by the Korean government MSIT: (No. 2022-0-01199, Graduate School of Convergence Security at Sungkyunkwan University), (No. 2022-0-01045, Self-directed Multi-Modal Intelligence for solving unknown, open domain problems), (No. 2022-0-00688, AI Platform to Fully Adapt and Reflect Privacy-Policy Changes), (No. 2021-0-02068, Artificial Intelligence Innovation Hub), (No. 2019-0-00421, AI Graduate School Support Program at Sungkyunkwan University), and (No. 2023-00230337, Advanced and Proactive AI Platform Research and Development Against Malicious deepfakes).

References

[Brown *et al.*, 2020] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[Cho and Hariharan, 2019] Jang Hyun Cho and Bharath Hariharan. On the efficacy of knowledge distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4794–4802, 2019.

[Deng *et al.*, 2009] 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.

[Dosovitskiy *et al.*, 2020] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

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

[He *et al.*, 2016] 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.

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

[Ji *et al.*, 2021] Mingi Ji, Seungjae Shin, Seunghyun Hwang, Gibeom Park, and Il-Chul Moon. Refine myself by teaching myself: Feature refinement via self-knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10664–10673, June 2021.

[Krizhevsky and Hinton, 2009] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical Report 0, University of Toronto, Toronto, Ontario, 2009.

[Lee *et al.*, 2022] Hanbeen Lee, Jeongho Kim, and Simon S Woo. Sliding cross entropy for self-knowledge distillation. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 1044–1053, 2022.

[Park *et al.*, 2019] 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*, pages 3967–3976, 2019.

[Passalis and Tefas, 2018] Nikolaos Passalis and Anastasios Tefas. Learning deep representations with probabilistic knowledge transfer. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 268–284, 2018.

[Ramesh *et al.*, 2021] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. *arXiv preprint arXiv:2102.12092*, 2021.

[Romero *et al.*, 2014] 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.

[Szegedy *et al.*, 2015] 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.

[Teerapittayanon *et al.*, 2016] Surat Teerapittayanon, Bradley McDanel, and Hsiang-Tsung Kung. Branchynet: Fast inference via early exiting from deep neural networks. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 2464–2469. IEEE, 2016.

[Tian *et al.*, 2019] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. *arXiv preprint arXiv:1910.10699*, 2019.

[Tung and Mori, 2019] Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1365–1374, 2019.

[Yang *et al.*, 2021] J Yang, B Martinez, A Bulat, G Tzimiropoulos, et al. Knowledge distillation via softmax regression representation learning. International Conference on Learning Representations (ICLR), 2021.

[Yim *et al.*, 2017] Junho Yim, Donggyu Joo, Jihoon Bae, and Junmo Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4133–4141, 2017.

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

[Zhang *et al.*, 2019] Linfeng Zhang, Jiebo Song, Anni Gao, Jingwei Chen, Chenglong Bao, and Kaisheng Ma. Be your own teacher: Improve the performance of convolutional neural networks via self distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3713–3722, 2019.

[Zhou *et al.*, 2021] Helong Zhou, Liangchen Song, Jiajie Chen, Ye Zhou, Guoli Wang, Junsong Yuan, and Qian Zhang. Rethinking soft labels for knowledge distillation: A bias-variance tradeoff perspective. *arXiv preprint arXiv:2102.00650*, 2021.