AlignKD: A Low-cost Technique for Convolutional Shortcut Removal

Shiu-hong KAO (20657378), Bo-Rong LAI (20737984)

Abstract

Deep neural networks have become increasingly robust due to the invention of the residual network. While adding shortcuts to networks has been widely used, the downside has also drawn more attention. Even though implementing residual networks significantly increases the accuracy, additional computational cost introduced by these shortcuts is frequently overlooked. Many studies have been working on the field of how to cut down the cost while maintaining the accuracy during training. In this paper, we have looked into several papers related to shortcut removal, and we especially investigated the approach using knowledge distillation. The approach adapts the teacher-student paradigm which reduces the inference time. After evaluating the performance of this strategy, we assert that operational redundancy exists inside the current teacher-student paradigm. In particular, multiple copies of feature maps are computed repeatedly in the teacher model. We further adjust it and develop our way to reintroduce the transferal approach from teacher to student. We proposed a novel approach AlignKD by making the distillation procedure aligned between the teacher and student model stage by stage. AlignKD is expected to save more computational cost for producing the repeated feature maps, thus are anticipated to see a shorter training time for the network to converge. During the experiment, we have undergone the datasets CIFAR10, CIFAR100, and Imagewoof for image classification. Finally, our proposed method achieves comparable or even higher results than the previous work with a lower time cost. However, several potential improvements are also included in the discussion.

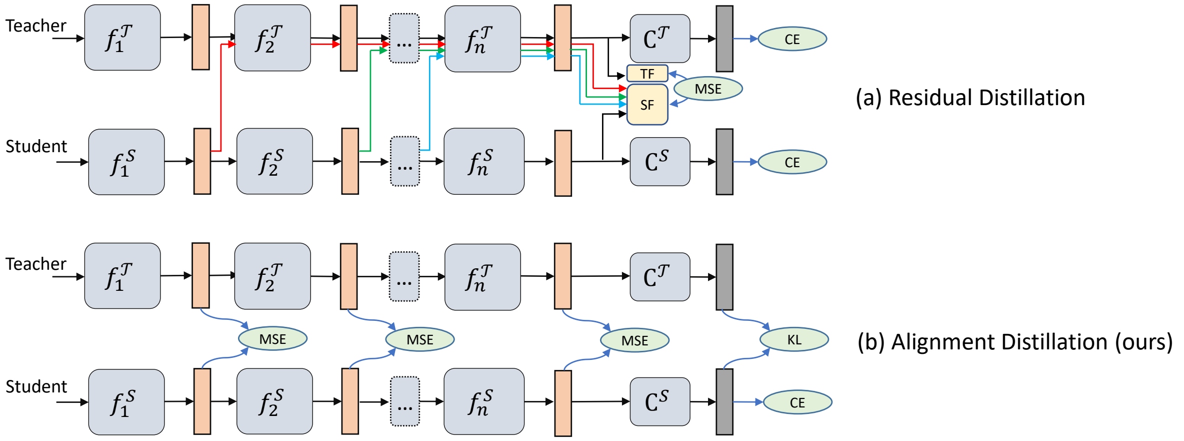


Figure 1. Methods to remove convolutional shortcuts

1. Introduction

In recent years, deep neural networks have improved dramatically with high accuracy in many applications. While increasing accuracy, the improvement of performance is often accompanied by the depth and capacity of the convolutional module. Among all the issues, the gradient vanishing problem has become widely discussed. The problem is invoked while entering relatively deeper layers during a training process. Thus, multiple state-of-the-art models have aimed to address this issue. Among all these studies, the introduction of shortcut architecture in ResNet [4] has gained remarkable popularity. Later on, plenty of papers such as MobileNet [5], ResNeXt [13], EfficientNet [11], etc., have also included shortcuts mechanism in their work. Usually, there are two common interpretations of shortcut utilization: preservation of previous features in the forward pass and transferring the gradient information during the backward pass.

However, the downside of implementing shortcuts has often been overlooked. According to Arash et al. [1], the shortcut mechanisms can account for nearly 40 percent of total feature maps. These massive data heavily occupied off-chip memory traffic consumption. To tackle the issue, there are two existing studies we have investigated. The first one is RepVGG [3], which suggested to remove shortcuts by combining the matrix operations via re-parametrization. After a series of mathematically equivalent transformation, multiple matrices are merged into one during test time for saving more computational power. The following one is JointRD [9]. In [9], they proposed a novel joint-training network which it aimed to train a naive CNN model by transferring the learned knowledge from a pre-trained ResNet counterpart model. This addressed the invoking of gradient vanishing problem by enabling the plain-CNN model with shortcuts feature from ResNet. The experiments had reported that the accuracy of plain-CNN model trained by JointRD can obtain comparable accuracy with pure ResNet model, and meanwhile were able to abridge the inference time.

While JointRD has gained great success in removing the shortcuts from the model, the computational cost for the removal process remains high. In this paper, we suggest a more streamlined architecture based on the proposal in [9]. Based on the original architecture for distillation, we utilized a different approach to divide the architecture into multiple stages. The main goal of our design is to avoid redundant computation during the distillation process. To transfer the knowledge directly from the intermediate features, we aligned the plain-CNN model with its counterpart model and compute feature-based loss after each stage. By doing this, we implicitly make the plain-CNN model to mimic the behavior of its counterpart as much as possible. Eventually the loss of rest layers are calculated by logit-based function. Together, we form an architecture enables a naive model the ability to acquire knowledge from a complex model without shortcuts implementation. Compare to the original work, our framework eliminate repeated forward passes in the teacher's model, thus are expected to increase the time efficiency. In addition, by conserving the original joint training paradigm, we prove it is still realizable to remove all the shortcuts while solving the gradient vanishing problem and preserving the previous features.

2. Related Work

According to [12], they suggested multi-branch topology, such as ResNet, can be considered as an implicit ensemble of numerous models. In [3], based on the proposal of [12] point out that the benefits of ensemble models are not desirable for inference. They further proposed the decoupling of the training time and inference-time architecture by utilizing structural re-parameterization, which means converting the original architecture into another form by transforming its parameters. The final model has a VGG-alike plain topology which only consists of a feed-forward process. However, trying to successfully merge two blocks by re-parameterizing requires the operations between the blocks to be linear. Consequently, unless the mathematical requirements for the matrix operation are satisfied, otherwise, such transformation can not be performed. This may not be practical in a real training scenario, since non-linearity layers are widely implemented among most state-of-the-art deep neural networks.

As for JointRD [9], it solved the aforementioned issue from a different approach. Based on the teacher-student paradigm, they make the student model be implemented without shortcuts both in the training and inference stage. Adapting a teacher-student paradigm, the JointRD make the teacher pass distilled information to student model. The teacher, a pre-trained ResNet model in the paper, particularly leveraged the gradient information the plain-CNN model. While forwarding the pass, the precedent layers of plain-CNN model will pass its outputs to the later stages of ResNet to calculate the loss. This can be interpreted as offering the model with auxiliary architecture to help it converge faster. During backpropagation, the ResNet model will give the naive student model mixture gradient computed with both models involved. However, since the there is a passing back to the teacher scenario after each layer. As going into deeper networks, there are multiple presence of copies of student outputs need to be processed by the teacher model. This may result in expensive computation. Unfold this mechanism, we suspect that the it may result in high redundancy inside the networks. In particular, we attempt to provide another solution that also contains training information from all layers, yet cost less computation to obtain.

3. Methodology

In this section, we will introduce and compare the methodology of the previous work [9] and our proposed framework. To simplify the notation, we define some variables for the hyper-parameters; in particular, we suppose the network is composed of n convolutional stages and one fully-connected stage, where $f_1^{\mathcal{T}}, f_2^{\mathcal{T}}, ..., f_n^{\mathcal{T}}$ represent the n convolutional stages of the teacher, $f_1^{\mathcal{S}}, f_2^{\mathcal{S}}, ..., f_n^{\mathcal{S}}$ imply the n convolutional stages of the student, and $C^{\mathcal{T}}, C^{\mathcal{S}}$ as the fully connected layer. Also, we define $\mathbf{z} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{b}$ as a batch of training data with size b in which $\mathbf{x}, \mathbf{y}$ represent the image and the label respectively. Let $f(\mathbf{x})$ imply the output after passing $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_b$ to the network f and $f \circ g(\mathbf{x}) = f(g(\mathbf{x}))$ for networks f and g, and $f_{i,j}^{N} = f_j^{N} \circ f_{j-1}^{N} \circ f_i^{N}$ for $i \leq j$ for $N \in \{\mathcal{T}, \mathcal{S}\}$. The loss term to update the student is denoted as $L_{\mathcal{S}}(\mathbf{z})$ for the training dataset $\mathbf{z}$.

3.1. JointRD

We refer to [9] as JointRD, demonstrating its methodology in this part. This method assumes that the teacher is well-pretrained before distillation and frozen during the back-propagation. The loss term of JointRD contains n feature-based losses and n logit-based losses. As shown in 1(a), the output of each stage of the student is passed to the deeper stages of the teacher network to generate the features. It defines

$$L_i^{CE}(\mathbf{z}) = \begin{cases} \frac{1}{b}\sum_{i=1}^{b} L_{CE}(C^{\mathcal{S}} \circ f_{1,n}^{\mathcal{S}}(\mathbf{x}), \mathbf{y}) & , \text{ if } \quad i=1 \\ \frac{1}{b}\sum_{i=1}^{b} L_{CE}(C^{\mathcal{T}} \circ f_{i,n}^{\mathcal{T}} \circ f_{1,i-1}^{\mathcal{S}}(\mathbf{x}), \mathbf{y}) & , \text{ if } \quad i=2,3,...,n \end{cases} \tag{1}$$

$$L_i^{mse}(\mathbf{z}) = \frac{1}{b}\sum_{i=1}^{b} L_{mse}(C^{\mathcal{T}} \circ f_{i+1,n}^{\mathcal{T}} \circ f_{1,i}^{\mathcal{S}}(\mathbf{x}), C^{\mathcal{T}} \circ f_{1,n}^{\mathcal{T}}(\mathbf{x})) \tag{2}$$

, where $L_{CE}(\cdot,\cdot)$ and $L_{mse}(\cdot,\cdot)$ indicates the cross-entropy loss and mean-square error correspondingly. In general, the final loss term can be represented as

$$L_{\mathcal{S}}(\mathbf{z}) = L_1^{CE}(\mathbf{z}) + \eta(\sum_{i=2}^{n} L_i^{CE}(\mathbf{z}) + \lambda \sum_{i=1}^{n} L_i^{mse}(\mathbf{z})) \tag{3}$$

, in which η, λ are two hyper-parameters for loss weight tuning.

We argue that this method involves several redundant operations due to its high computational cost, especially sending the data from the student to the teacher, since the i-th stage in the teacher is computed i times by this design. We hence propose an adjusted approach called *Alginement Distillation* to reduce the cost.

3.2. Alignment Distillation (AlignKD)

To abridge the time of shortcut removing process, one of the most significant ways is to avoid unnecessary operations in JointRD. Here we propose Alignment Distillation (AlignKD) as a new framework for shortcut removal and present the design in 1(b). Instead of measuring the feature loss in the final convolutional layer before the fully-connected one, we calculate the loss between the two stages directly using their outputs. In AlignKD, the loss term is composed of n feature-based losses and two logit-based losses. The feature losses are defined in such a way:

$$L_i^{mse}(\mathbf{z}) = \frac{1}{b}\sum_{i=1}^{b} L_{mse}(f_{1,i}^{\mathcal{S}}(\mathbf{x}), f_{1,i}^{\mathcal{T}}(\mathbf{x})) \tag{4}$$

Besides, two logit-based loss term are defined as

$$L^{KL}(\mathbf{z}) = \frac{1}{b}\sum_{i=1}^{b} L_{KL}(C^{\mathcal{S}} \circ f_{1,n}^{\mathcal{S}}(\mathbf{x}), C^{\mathcal{T}} \circ f_{1,n}^{\mathcal{T}}(\mathbf{x})) \tag{5}$$

$$L^{CE}(\mathbf{z}) = \frac{1}{b}\sum_{i=1}^{b} L_{CE}(C^{\mathcal{S}} \circ f_{1,n}^{\mathcal{S}}(\mathbf{x}), \mathbf{y}) \tag{6}$$

, where $L_{KL}(\cdot,\cdot)$ implies the Kullback–Leibler divergence between two provided elements. The final loss term is then modified as

$$L_{\mathcal{S}}(\mathbf{z}) = \alpha L^{KL}(\mathbf{z}) + (1-\alpha) L^{CE}(\mathbf{z}) + \eta \sum_{i=1}^{n} L_i^{mse}(\mathbf{z}) \tag{7}$$

, in which α is a hyper-parameter for the weight of logit-based KD loss and η is a hyper-parameter for the feature-based KD loss. In practice, we set $alpha$ to 0.5 and η to be cosine-annealing decay from 1 to 0.5.

4. Datasets

During training, three datasets, CIFAR-10 [8], CIFAR-100 [8] and Imagewoof [6], are mainly used for the experiments. CIFAR-10 and CIFAR-100 have 10 classes and 100 classes respectively with a small scale of images. Both of them contain 50,000 training images and 10,000 testing images, widely used as the baselines for image classification. Imagewoof, on the other hand, provides a larger scale of images and has a smaller training dataset. It contains 10 dog classes sampled from ImageNet [2], usually regarded as a more challenging task than CIFAR-10.

5. Experiments

In this paper, experiments on three datasets, CIFAR-10 [8], CIFAR-100 [8], and Imagewoof [6], are conducted to compare the effectiveness and efficiency of our proposed framework, AlignKD, and JointRD. In JointRD, the same hyper-parameters setting was used to maintain the fairness, where η was cosine-annealing decreasing from 1.0 to 0.5 in the first 60 epochs and λ was set to 0.001. In AlignKD, the hyper-parameter α was set to 0.5 and the η decreases annealing with the same strategy as JointRD. In these experiments, we define *Plain-CNN* as the shortcut-removed network with the same architecture as ResNet. Each experiment was conducted in 200 epochs with a learning rate 0.01 and OneCycle learning rate scheduler [10], where the max learning rate was set 10 times the optimizer's learning rate. ResNet18 and ResNet34 are the two models considered for shortcut removal. Appendix A provides the validation accuracy for the experiments in this section.

5.1. CIFAR-10

In this subsection, vanilla ResNet18 and ResNet34 were used for the shortcut-removing experiments. Due to the small scale of CIFAR images, the max-pooling layer followed by the first convolutional layer was removed for better performance. This technique is commonly used for the ResNet implementation on the CIFAR dataset.

JointRD	Acc (%)	Time (mins)
Plain-CNN 18	92.99	107
Plain-CNN 34	92.41	178
AlignKD	Acc (%)	Time (mins)
Plain-CNN 18	92.97	94
Plain-CNN 34	92.82	152

Table 1. CIFAR-10 Shortcut Removal (ResNet18: 93.57, ResNet34: 93.86). Plain-CNN18 and Plain-CNN34 are the models removing shortcuts from ResNet18 and ResNet34 respectively. Each experiment was conducted using a pre-trained ResNet as the teacher and a Plain-CNN as the student, where these two models have the same depth.

We first trained these two ResNet's naively with 200 epochs, and regard these models as the teachers in the following experiments. In specific, ResNet18 and ResNet34 obtained accuracy of 93.57% and 93.86% respectively. To remove the shortcuts from the ResNet, the Plain-CNN network with the same depth were adopted as the students. The experimental results are shown in Table 1. By comparing the results of the same model with two different removing strategies, we suggest that AlignRD generally achieves comparable results with lower costs than JointRD.

5.2. CIFAR-100

JointRD	Acc (%)	Time (mins)
Plain-CNN 18	76.68	133
Plain-CNN 34	74.89	222
AlignKD	Acc (%)	Time (mins)
Plain-CNN 18	76.85	117
Plain-CNN 34	74.83	190

Table 2. CIFAR-100 Shortcut Removal (ResNet18: 77.25, ResNet34: 78.39). Plain-CNN18 and Plain-CNN34 are the models removing shortcuts from ResNet18 and ResNet34 respectively. Each experiment was conducted using a pre-trained ResNet as the teacher and a Plain-CNN as the student, where these two models have the same depth.

Analogous to the model design in the CIFAR-10 experiments, all the max-pooling layers in the models for CIFAR-100 were also replaced with identity layers. The pre-trained ResNet18 and ResNet34 had accuracy of 77.25 and 78.39 accordingly. As shown in Table 2, AlignKD outperformed JointRD in the experiment of removing shortcuts from ResNet18, and the training time cost was also lower. It also demonstrated a comparable performance when a larger network was considered.

JointRD	Acc (%)	Time (mins)
Plain-CNN 18	80.21	50
Plain-CNN 34	77.11	76
AlignKD	Acc (%)	Time (mins)
Plain-CNN 18	79.4	44
Plain-CNN 34	77.84	64

Table 3. Imagewoof Shortcut Removal (ResNet18: 80.86, ResNet34: 82.66). Plain-CNN18 and Plain-CNN34 are the models removing shortcuts from ResNet18 and ResNet34 respectively. Each experiment was conducted using a pre-trained ResNet as the teacher and a Plain-CNN as the student, where these two models have the same depth.

5.3. Imagewoof

In previous subsections, we presented the effectiveness of AlignKD on small-scaled datasets. It has shown success in abridging the training time for shortcut removal while preserving the performance. We argue that a more difficult task is needed for the generality of our proposed framework. Imagewoof, a 10-class subset of ImageNet, contains images with higher resolutions than the CIFAR datasets. The number of training samples per class is also smaller than the aforementioned datasets.

Instead of removing the max-pooling layer, we maintained this layer in ResNet architecture and conducted the same experiments on Imagewoof. Table 3 shows that the results are also promising, encouraging us to explore the further possibilities for AlignKD.

6. Discussion

Since our method intentionally makes the student stage mimic the behavior of its corresponding stage from the teacher instead of transferring the gradient knowledge, we argue that the cross-model inconsistency can be reduced. A comparable or even better performance from AlignKD can be expected. Empirically, from section 5, the AlignKD sometimes outperformed the JointRD method by a small margin in terms of accuracy. Also from Tables 1 2 3, these data also illustrate the advantages that the AlignKD converges faster than JointRD. This can be attributed to the removal of multiple inputs passed from the student to the teacher, hence both redundant computation and execution time can be saved.

While AlignKD has gained success in reducing the training time cost, the distinction may be remaining improved. According to the experimental results, we suggest that the cost reduction was not as significant as expected. For example, in Table 3, the reduced time of training Plain-CNN18 between the JointRD approach and our method is approximately 6 minutes only. We preliminarily ascribe the unexpected result to the expensiveness of computing MSE loss. In AlignKD, the equation 4 computes the MSE loss in the middle layers of teacher and student, where the input tensors tend to be highly dimensional. Compared to the JointRD, the equation 4 only takes into the flattened features from the last layer. We deduce that the computational cost in terms of different dimension matters more than we thought before. As a result, the performance was not as significant as we have pictured. To improve this, we provide several approaches to distill feature-based knowledge which can be considered helpful, including Attention Transfer [14] and Neuron Selectivity Transfer [7].

Furthermore, even though the inference time is faster without shortcuts, we can see that the performance of Plain-CNN34 is statistically not as good as that of Plain-CNN 18. One possible interpretation is that Plain-CNN 34 is a relatively deeper network, and deeper networks are normally considered harder to train. Removing the shortcuts and applying knowledge distillation may still result in insufficient accuracy.

7. Conclusion

We have proposed a novel architecture based on a knowledge distillation strategy that can achieve shortcut removal while maintaining the accuracy. From the experiment, we can see the results are roughly aligned with our expectations in the positive direction. We have successfully implemented AlignKD which makes the student align with the teacher model in a stage-by-stage strategy. However, we also discovered some flaws in our approach which possibly is the main reason why our model outperformed the JointKD marginally in terms of time. Moreover, a declination in accuracy has also been observed as our student model went deeper. In the future, in order to further increase the time efficiency, finding another low-cost approach for knowledge distillation to effectively train the shortcut-removed networks is required. Finally, we would also like to explore the potential of our proposed method by transferring knowledge from a larger model to a smaller one; for example, an experiment using ResNet34 as teacher and Plain-CNN18 as student also draws our attention, since the success of this experiment can provide us with a more efficient network by compressing the model depth.

References

[1] Arash Azizimazreah and Lizhong Chen. Shortcut mining: Exploiting cross-layer shortcut reuse in dcnn accelerators. In *2019 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pages 94–105. IEEE, 2019. 2

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 3

[3] Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. Repvgg: Making vgg-style convnets great again. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13733–13742, 2021. 2

[4] 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

[5] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 1

[6] Jeremy Howard. Imagewang. 3

[7] Zehao Huang and Naiyan Wang. Like what you like: Knowledge distill via neuron selectivity transfer. *arXiv preprint arXiv:1707.01219*, 2017. 5

[8] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 3

[9] Guilin Li, Junlei Zhang, Yunhe Wang, Chuanjian Liu, Matthias Tan, Yunfeng Lin, Wei Zhang, Jiashi Feng, and Tong Zhang. Residual distillation: Towards portable deep neural networks without shortcuts. *Advances in Neural Information Processing Systems*, 33:8935–8946, 2020. 2, 3

[10] Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial intelligence and machine learning for multi-domain operations applications*, volume 11006, pages 369–386. SPIE, 2019. 3

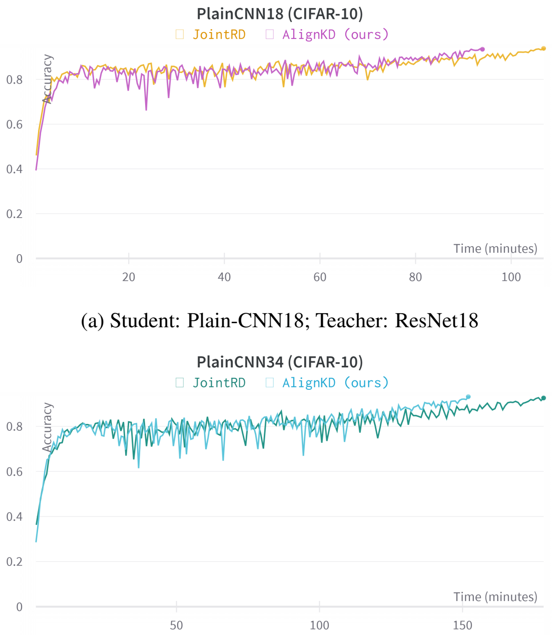
[11] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019. 1

[12] Andreas Veit, Michael J Wilber, and Serge Belongie. Residual networks behave like ensembles of relatively shallow networks. *Advances in neural information processing systems*, 29, 2016. 2

[13] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017. 1

[14] 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. 5

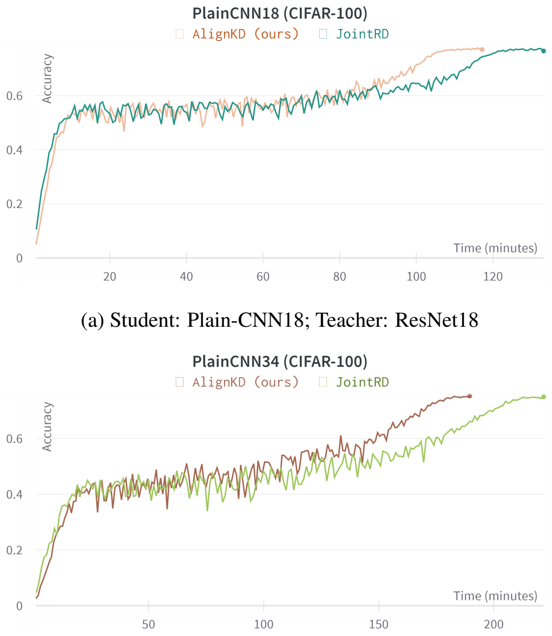
A. Experimental Charts



(a) Student: Plain-CNN18; Teacher: ResNet18

(b) Student: Plain-CNN34; Teacher: ResNet34

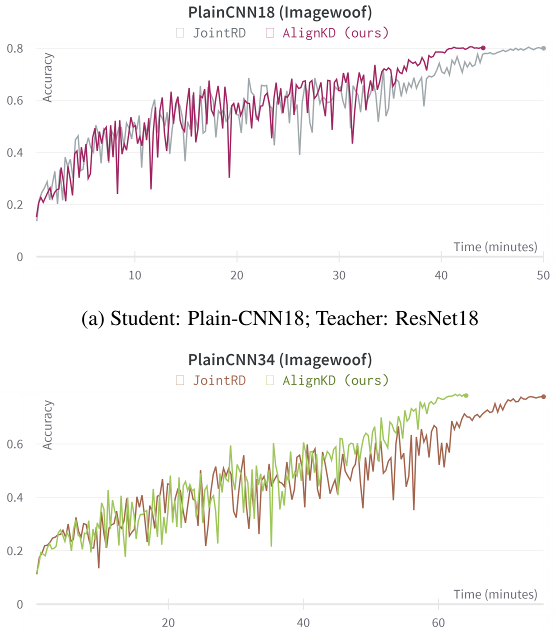
Figure 2. CIFAR-10 validation accuracy



(a) Student: Plain-CNN18; Teacher: ResNet18

(b) Student: Plain-CNN34; Teacher: ResNet34

Figure 3. CIFAR-100 validation accuracy



(a) Student: Plain-CNN18; Teacher: ResNet18

(b) Student: Plain-CNN34; Teacher: ResNet34

Figure 4. Imagewoof validation accuracy