entropy

MDPI

Article

Multistructure-Based Collaborative Online Distillation

Liang Gao [1], Xu Lan [2], Haibo Mi [1], Dawei Feng [1], Kele Xu [1,*] and Yuxing Peng [1]

[1] National Key Laboratory of Parallel and Distributed Processing, College of Computer, National University of Defense Technology, Changsha 410073, China; gaoliang13@nudt.edu.cn (L.G.); rainmhb@gmail.com (H.M.); davyfeng.c@gmail.com (D.F.); pengyuhang@nudt.edu.cn (Y.P.)

[2] School of Electronic Engineering and Computer Science, Queen Mary University of London, London E14NS, UK; x.lan@qmul.ac.uk

* Correspondence: kelele.xu@gmail.com; Tel.: +86-166-7316-1118

Received: 5 March 2019; Accepted: 1 April 2019; Published: 2 April 2019

Abstract: Recently, deep learning has achieved state-of-the-art performance in more aspects than traditional shallow architecture-based machine-learning methods. However, in order to achieve higher accuracy, it is usually necessary to extend the network depth or ensemble the results of different neural networks. Increasing network depth or ensembling different networks increases the demand for memory resources and computing resources. This leads to difficulties in deploying depth-learning models in resource-constrained scenarios such as drones, mobile phones, and autonomous driving. Improving network performance without expanding the network scale has become a hot topic for research. In this paper, we propose a cross-architecture online-distillation approach to solve this problem by transmitting supplementary information on different networks. We use the ensemble method to aggregate networks of different structures, thus forming better teachers than traditional distillation methods. In addition, discontinuous distillation with progressively enhanced constraints is used to replace fixed distillation in order to reduce loss of information diversity in the distillation process. Our training method improves the distillation effect and achieves strong network-performance improvement. We used some popular models to validate the results. On the CIFAR100 dataset, AlexNet's accuracy was improved by 5.94%, VGG by 2.88%, ResNet by 5.07%, and DenseNet by 1.28%. Extensive experiments were conducted to demonstrate the effectiveness of the proposed method. On the CIFAR10, CIFAR100, and ImageNet datasets, we observed significant improvements over traditional knowledge distillation.

Keywords: deep learning; knowledge distillation; distributed architecture; supplementary information

1. Introduction

The development of deep learning [1,2] has led to a leap in the fields of computer vision [3–6] and natural language processing [7–11]. In image recognition in particular [4,12,13], recognition accuracy has reached a high level by using deep-learning methods. However, high-quality models are often accompanied by huge parameter quantities, huge computing resources, and huge storage requirements [3,14,15]. The huge demand for resources is an important obstacle to the promotion and use of deep-learning models in the industry. Especially in resource-preferred scenarios such as FPGA, mobile devices, and microcomputers, the contradiction between performance improvement and resource occupancy is more intense. Traditionally, training a deeper network or merging multiple models [16,17] may achieve better performance, but this cannot avoid the growth of resource consumption. The problem of how to improve performance without increasing network size has received extensive attention. Some training methods, such as model compression [18,19] and model pruning [20], have been proposed to solve this problem.

The distillation [21,22] method (that is, teacher–student training) is effective to solve the contradiction between network scale and network accuracy. Distillation is mainly used for classification problems. Ground-truth labels in supervised learning problems are usually of the one-hot type, only focused on the truth category. The basic idea of distillation is to extract the subcategory information of the network. Although forcing the classification of the sample to the ground-truth label is effective, it is not necessarily optimal, as it ignores the similarity information of samples in categories. Learning a similarity matrix [23,24] can preserve the similarity between classes, but sample characteristics are neglected. Category-similarity information for different samples is different. In knowledge distillation, a complex large network called 'teacher' extracts sample-level category-similarity knowledge (sample classification probability P); then, students (simple network) use it as their training targets. The teacher calculates class probability P for each sample and uses it to guide students' learning. Knowledge distillation reduces the difficulty of students' optimization. The student network that cannot learn complete interclass similarity due to its structural constraints would be better optimized by learning from the teacher. Compared with the ground-truth label, class probability contains the samples' similarity knowledge in all categories, which enhances the effect of students' learning.

Traditional knowledge is static and two-stage, which enhances the performance of the student network, but it is only useful for small networks that perform poorly. However, the performance improvement of large networks is more meaningful and difficult. The deep mutual-learning (DML) method [25] uses a group of students to improve their performance by learning from each other. Students continuously learn from other's classified probability so that each student can maintain the same class probability as others. Every student is better than traditional supervised learning. In DML, several large networks are promoted from each other, but there are still shortcomings: continuous mutual imitation weakens the generation of complementary information so that it reduces the final generalization ability, the number of students is limited by the single machine's resources, and it is difficult to achieve effective expansion.

In this paper, we propose a multistructural model online-distillation method. Compared with other work, our method not only has stronger compatibility (multistructure), scalability (distributed environment), but also better performance (higher-precision improvement). Our method is mainly based on two premises: a stronger teacher educates better students, and appropriate distillation methods can reduce information loss.

Better teachers are made up of differentiated students. In knowledge distillation, a good teacher determines the upper limit of the student model. We adopted three strategies to strengthen the teacher model. We trained a group of students under distributed conditions. Using the weighed average ensemble method [16,17], we summarized student models' information to form a teacher model. The ensemble effect is mainly influenced by the complementary knowledge between the student models. We extended the distillation method to a distributed framework so that we could accommodate more student models to help the teacher reduce the risk of overadaptation. In addition, students of different structures have more complementary information; we used soft labels as the information-exchange medium to jointly train networks of different structures.

Gradually intensifying knowledge distillation reduce information loss. For continuous mutual imitation, there is a risk of information consistency. We used interval distillation to increase the information diversity of the student model by inserting independent training. Although the use of interval distillation enhances overall information growth, it faces loss-function switching, and sudden changes may result in the loss of network information. We gradually increased teacher constraints so that students' loss became flat to avoid information loss. These methods reinforce the feedback between teachers and students, and increase information complementarity. Our results go beyond the previous distillation method.

The shortcoming of our method is that multiple student nodes lead to greater training overhead and greater information redundancy. Fortunately, there was no change in time complexity and space complexity when used, and better performance than previous methods was achieved. In general,

multistructure online distillation enhances information diversity in the distillation process and improves the accuracy of the model through multinetwork cooperation. Extensive experimentation was carried out on image-recognition issues using popular network architectures and achieved the highest performance improvement. Our approach has the following advantages:

1. Effectively utilizes diversity between different structural models.
2. Demand for network resources is low and applicability is stronger.
3. Network performance achieved the highest improvement while not increasing resource occupation.

The organization of this paper is as follows. Section 2 briefly reviews the related work, and Section 3 describes our approach. The experiment is shown in Section 4, and we summarize this work in Section 5.

2. Related Work

There is a lot of work to improve the performance of deep-learning models through knowledge transfer. This section describes some of the network-interaction methods that are relevant to our work.

In the study of network interaction, some distributed methods [26–30] are used to accelerate the training of the network. For example, the parameter average MA [28–30] method and the distributed stochastic gradient descent algorithm [31] are widely used to accelerate the training process through multinode information exchange. In addition, multitasking learning [32–35] mainly plays a role in feature selection to lift accuracy. In multitask learning, there are multiple training objectives at the same time. The network has different characteristics for feature selection based on different targets, forcing the network to learn complementary features to improve network performance. The ensemble method integrates the output of multiple networks to generate more flat predictions than a single network, improving the generalization of the model. There are many studies integrating multiple models, such as separated score integration (SSI) [36], Bayesian model averaging [37], and score fusion based on alpha integration [38]. These methods inspired us to design our own training framework.

Our approach is based on knowledge distillation [21,22,25,39,40]. Hinton, Geoffrey proposed knowledge distillation [22] that allows small models to learn the knowledge in a pretrained large model. The main motivation for distillation is that the teacher model's soft label provides knowledge of interclass similarity that cannot be provided by the ground-truth goal. The traditional method of knowledge distillation is limited by the direction of information flow, so it cannot improve performance on large networks. Lan, Zhu, and Gong improved knowledge distillation and proposed distillation method ONE [41], which uses a set of multibranch student models to learn from each other. Mutual imitation to enhance the learning of the target network, but ONE shares the low-level features lacking diversified feature learning. Codistilling [39] online distillation through large-scale distributed training can accommodate more nodes for training and accelerates the training process.

Existing distillation methods lack the inclusiveness of the network structure, which limits the complementary information that the structure can provide. In addition, we found that the distillation process forced multiple models to fully adapt to uniform soft-label output, which actually weakened the diversity of student development, which was detrimental to the end result.

In this paper, we created a distributed cross-structure online-distillation method and loosened distillation constraints to enhance network diversity. We used soft labels that are independent of the network structure as the information-exchange medium, abandoning the parameters and gradients commonly used for traditional distributed-information exchange. Soft labels meet the needs of information exchange during the distillation process and are structurally independent, allowing our students to combine multiple structural models. The ensemble method is helpful to form teachers who are better than previous knowledge-distillation methods. Considering that continuity distillation leads to the simplification of model information, we used interval distillation to enhance model diversity, and gradually enhanced distillation to capture small information differences.

3. Methodology

This section introduces our methods. First, in the overview, we introduce our methods and principles. Then, details of the implementation of the method are introduced in the following section.

3.1. Overview

As Figure 1 shows, we used multiple networks for distributed joint training; each network is a student. The teacher network is the knowledge ($\widehat{S}_e$) aggregated by all student networks. Training is divided into two stages, independent training and distillation. In the independent-training phase, the model learns the relationship between training data and the ground-truth label by minimizing cross-entropy loss. After a certain independent-training period, students' soft labels are uploaded to the server. In the server, we aggregated student information by calculating the weighted average values of their soft labels ($\widetilde{S}_1, \widetilde{S}_2, ..., \widetilde{S}_N$). Then, the aggregated soft labels ($\widehat{S}_e$) were sent to each student to guide students' distillation training. Then, in the distillation process students are optimized by minimizing Kullback–Leibler divergence between fixed aggregated soft labels $\widehat{S}_e$ and students' soft labels $\widetilde{S}_i$. After distillation is finished, we return to the independent-training stage.

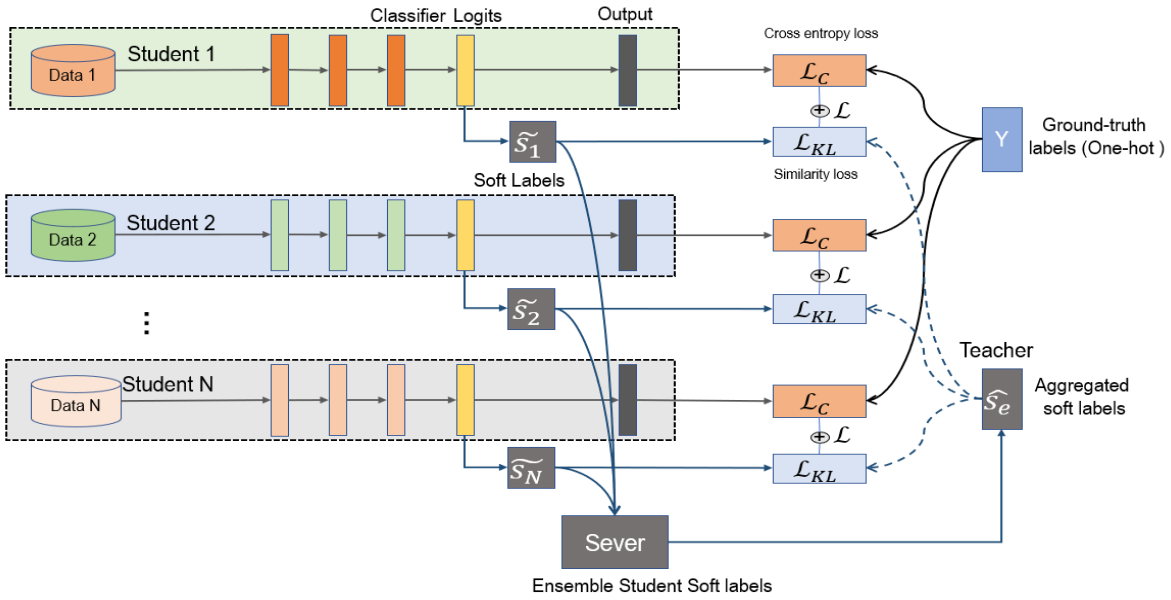
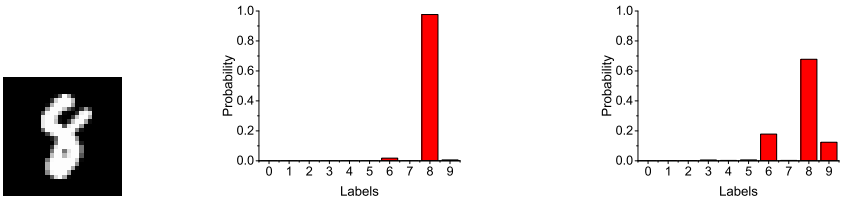


Figure 1. Overview of our approach. Multiple students work together, each student is a separate model, and the teacher was the aggregated information ($\widehat{S}_e$) from multiple student networks. First, students train their models by minimizing cross-entropy loss $\mathcal{L}_C$ to learn from ground-truth label Y. Then, the server aggregates student information ($\widetilde{S}_1, \widetilde{S}_2, ..., \widetilde{S}_N$) to generate teacher information ($\widehat{S}_e$). Finally, the teacher feeds back the student by minimizing Kullback–Leibler divergence $\mathcal{L}_{KL}$ between fixed aggregated soft labels $\widehat{S}_e$ and student soft labels $\widetilde{S}_i$ during the distillation phase.

Distributed Training. We distributed training a group of students and, by regularly aggregating information in the server, we used the aggregated information as the teacher in distillation. Soft labels were used for information transmission instead of parameters or gradients. Soft labels have network-independent characteristics, so we could mix networks of different structures. Soft labels do not need to be updated at every step, longer transmission interval is allowed. Our distributed training has three advantages: reducing network overhead, compatibility with different network structures, and expansion-capacity enhancement.

Ensemble gets better teachers. The distillation method improves network performance because of supplementary information. Different network structures and different initializations can generate different knowledge. The model expresses the relationship between data and labels by probability

output. Soft labels (as shown in Figure 2) are the softening probabilities that not only concern sample classification, but also sample similarity. The similarity relationships provide additional information during distillation. We collected soft labels from all students and ensembled them by calculating the weighted average values. The aggregated soft labels had better generalization performance, abstracted as the teacher in distillation.



(a) input (b) Standard probability. (c) Soft label probability.

Figure 2. On the MNIST dataset, Alexnet after 50 rounds of training outputs. (**a**) One sample from MNIST handwritten digit database, (**b**) standard probability (temperature = 1) and (**c**) soft-label probability (temperature = 3) for input. Soft labels can express the class-similarity relation more comprehensively.

Better distillation. Additional knowledge is important for distillation. Compared with the previous method, we used interval distillation to expand information diversity. In our approach, the teacher is generated through the aggregation of students. The teacher has more supplementary information, while the students are more diverse. The paradox is that all students learn from the same teacher, which would make students' knowledge consistent and lack complementarity. We joined an independent-training phase without distillation constraints, so that students could learn independently and have diversified development. Switching the loss function between independent training and distillation causes the gradient value to change dramatically, which may lead to loss of model information. We gradually increased the limitation of teacher guidance (as Equation (10) shows) in the distillation process to retain more details.

The training process is shown in Figure 3. Our approach is a multicycle process. Each cycle can be roughly divided into five steps. These are independent training, distillation training, and information, transmission, reception, aggregation.

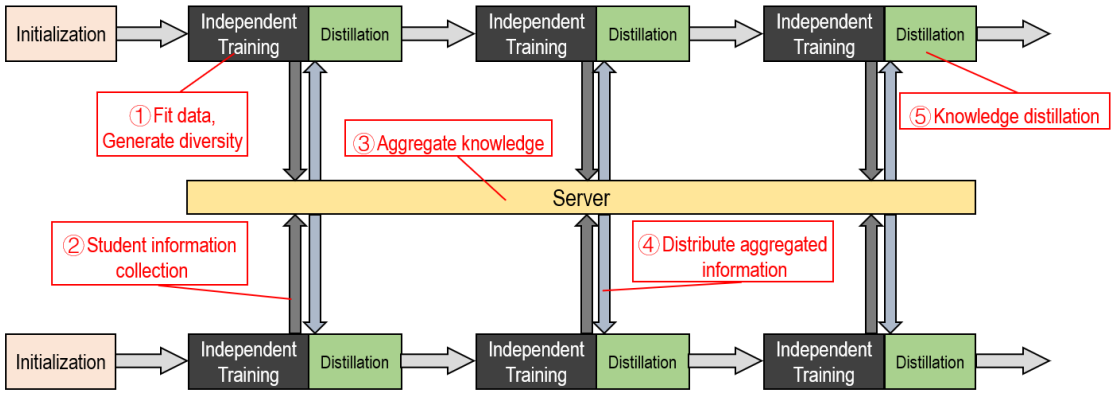


Figure 3. Our method's training process.

In this paper, we propose a cross-architecture online-distillation approach that improves classification accuracy without changing the network structure. We combined distillation and independent training in a cycle. The weighted average ensemble method was used to synthesize teacher knowledge; information on multistudents was extracted and utilized. The teacher guides students training through knowledge distillation. Student performance improved through three

stages: learning from local data distribution, generating teacher information by aggregating students' information, and learning from the teacher by distillation.

3.2. Independent Training

For a C class-classification task, assume there are N samples of input $\mathbb{X} = \{x_i | i \in (1, 2, ..., N)\}$, and the corresponding ground-truth labels $\mathbb{Y} = \{y_i | i \in (1, 2, ..., N)\}$. In general, ground-truth label y_i is a one-hot vector with a dimension of $1 \times C$, y_i is equal to 1 only in the position of the category to which sample x_i belongs, and equal to 0 in other positions. We take the i-th student as an example to introduce the training steps.

The student networks optimize the model by minimizing cross-entropy loss between predicted value p_i and ground-truth label y_i during the independent-training process. Students do not have information interaction, which is conducive to increasing the diversity of model information. Students' loss function could be freely chosen according to their respective situations. Here, we take the cross-entropy-loss function as an example:

$$\mathcal{L}_C(\mathbb{Y}|P) = -\frac{1}{N}\sum_{i=0}^{N-1}\sum_{k=0}^{C-1} y_i^k * log p_i^k \tag{1}$$

where p_i^k represents the normalized probability that a student model F_s classifies input x_i as the k-th class.

Logits is the output of the penultimate layer of the convolutional neural networks (CNN) used to classify the problem. In the CNN, input x_i is subjected to feature processing to obtain a feature vector $f(x_i)$ (assuming the $f(x_i)$ dimension is $1 \times m$). Then, we used feature-classification layers (linear layer) to map the feature space to the sample mark space by linear transformation, and the output is logits. The linear relationship between logits g_i and feature $f(x_i)$ can be abstracted as $g_i = f(x_i) * w + b$ (where dimension of w is $m \times C$, C is the number of classes), and logits g_i is a vector of dimension $1 \times C$. In general, logits g_i is normalized by the softmax layer to obtain classification-probability output p_i. Assume that logits were obtained by inputting x_i of the student network F_s is $[g_i^1, g_i^2, ..., g_i^C]$. Use the following formula to calculate normalized classification probability p_i^k.

$$p_i^k = \frac{exp(g_i^k)}{\sum_{k=1}^{C} exp(g_i^k)} \tag{2}$$

Normalized probability p_i characterizes the model's confidence in the classification decision of x_i. The closer the p_i^k value is to 1, the higher the confidence of x_i belonging to the k-th class.

To optimize the model parameters from θ_t to θ_{t+1} at the t-th iteration, we minimized loss with a back-propagating algorithm. The model was optimized as follows:

$$\theta_{t+1} = \theta_t - \eta * \frac{\partial \mathcal{L}_C(y_t | F_s(\theta_t, x_t))}{\partial \theta_t} \tag{3}$$

where η is the learning rate, (x_t, y_t) is the input data of the i-th iteration, $\frac{\partial \mathcal{L}_C(y_t | F_s(\theta_t, x_t))}{\partial \theta_t}$ is the partial derivative gradient of cross-entropy loss $\mathcal{L}_C(y_t | F_s(\theta_t, x_t))$ to θ_t.

Independent training in each cycle lasts for a $\mathbb{T}_{in}$ epoch. As the loss of $\mathcal{L}_C$ decreases, the student model fits the data distribution more accurately, and the more likely the sample is to be mapped to the correct classification. In the independent-training phase, student models do not communicate with each other, and they are more likely to fall into different local optimums. Student models have inconsistent information that can produce more complementary information.

3.3. Information Aggregation

We used soft labels as the medium of information. Soft labels are a softer normalized probability value that is more gradual and emphasizes the relationship between samples and classes. The soft-label calculation method is as follows:

$$\widetilde{p}_i^k = \frac{exp(g_i^k/t)}{\sum_{k=1}^{C} exp(g_i^k/t)} \tag{4}$$

Parameter t is the temperature parameter that is used to increase the degree of relaxation of the soft label and emphasis on the secondary category. However, a too-large t causes confusion in the category, and we chose $t = 3$ in our experiment.The soft-label aggregation corresponding to all inputs $\mathbb{X}$ is recorded as $\widetilde{S}, \widetilde{S} = \{\widetilde{p}_0, \widetilde{p}_1, ..., \widetilde{p}_{N-1}\}$.

At the end of the independent-training period, student model F_{si} calculates its own soft labels $\widetilde{S}_i$ generated for input $\mathbb{X}$, obtains a new data relationship $(\mathbb{X}, \widetilde{S}_i)$, and then uploads $\widetilde{S}_i$ to the server.

Using a soft label to calculate cross-entropy loss:

$$\mathcal{L}_C(\mathbb{Y}|\widetilde{S}) = -\frac{1}{N}\sum_{i=0}^{N-1}\sum_{k=0}^{C-1} y_i^k * log\widetilde{S}_i^k \tag{5}$$

Loss function $\mathcal{L}_C(\mathbb{Y}|\widetilde{S})$ is convexly optimized for soft-label value $\widetilde{S}$. For the nature of convex optimization, the weighted combination of any two students' cross-entropy loss has the following characteristics:

$$\mathcal{L}_C(\mathbb{Y}|(a * \widetilde{S}_1 + b * \widetilde{S}_2)) <= a * \mathcal{L}_C(\mathbb{Y}|\widetilde{S}_1) + b * \mathcal{L}_C(\mathbb{Y}|\widetilde{S}_2) \tag{6}$$

Simple promotion to any number of student models:

$$\mathcal{L}_C(\mathbb{Y}|\sum_{i=0}^{M} a_i * \widetilde{S}_i) <= \sum a_i * \mathcal{L}_C(\mathbb{Y}|\widetilde{S}_i) \tag{7}$$

We used the weighted average approach to aggregate student information. The weighted average method is equally simple compared to the average method, but the weights can be set based on model performance to reduce the impact of interference information. Using the weighted average method can ensure that the overall cross-entropy loss of student models reducing (according to Equation (7) shows), the information of the student is effectively integrated to reduce overfitting. The students' score weighted average process is as follows:

$$\widehat{S}_e = \sum_{i=0}^{M} a_i * \widetilde{S}_i \tag{8}$$

In the formula, $\widetilde{S}_i$ are the soft labels of i-th students, M is the student number, a_i is the weight, $\widehat{S}_e$ are the aggregated soft labels. Weighted value a_i of the student's soft labels is determined by the accuracy: $a_i = \frac{v_i}{\sum_{k=1}^{M} v_i^k}$. Where M is the number of student models and v_i is the model accuracy of student model F_{si}.

3.4. Distillation Fusion

In the distillation-fusion phase, aggregated soft labels (as teacher knowledge) $\widehat{S}_e$ are used to guide student training. Minimizing the Kullback–Leibler divergence of $\widehat{S}_e$ and $widetildeS_i$ to urge students to learn from the teacher. Kullback–Leibler divergence describes the discrepancy between

students' soft-label distribution $widetildeS_i$ and the teacher's distribution $\widehat{S}_e$. The Kullback–Leibler divergence-loss calculation formula of the i-th student is as follows:

$$\mathcal{L}_{KL}(\widehat{S}_i|\widetilde{S}_e) = -\frac{1}{N} \sum_{(\widetilde{p}_i^k, \widehat{p}_i^k) \in (\widetilde{S}_e, \widehat{S}_i)} \widetilde{p}_i^k * log \frac{\widetilde{p}_i^k}{\widehat{p}_i^k} \tag{9}$$

$\widetilde{p}_i^k$ is the soft-label probability of $\widetilde{S}_e$ for input sample x_i on the k-th class, and $\widehat{p}_i^k$ is the soft-label probability for the x_i in the k-th class on soft-label $\widehat{S}_i$ of the i-th student. $\widetilde{S}_e$ is the teacher model information that is sent to the students by the server in the previous step. $\widetilde{S}_e$ is fixed in the distillation-fusion stage, and $\widehat{S}_i$ is the soft-label output of the student model, updated with the update of the model parameters.

Kullback-Leibler divergence combines with standard cross-entropy loss $\mathcal{L}_C$ to maintain the target of the ground-truth label value. We used a weighted approach to balance the proportion of Kullback-Leibler divergence loss and cross-entropy loss. The loss function of the i-th student in the distillation-fusion phase is as follows:

$$\mathcal{L}_i = \alpha * \mathcal{L}_C^i + \beta * \mathcal{L}_{KL}^i \tag{10}$$

The weights of cross-entropy loss and Kullback-Leibler divergence loss in the distillation-fusion process are α and β. We gradually changed them other than using constant values, which we call "gradual knowledge transfer". Specifically, gradually reducing the weight of cross entropy while gradually increasing Kullback–Leibler divergence to achieve smooth loss transition in distillation.

$$\alpha = 1 - \frac{r}{\mathbb{T}_d} \qquad \beta = 1 + \frac{r}{\mathbb{T}_d} \tag{11}$$

Here, r is the epoch number of the distillation, and r is set to 0 at the start of each training cycle. The work of distillation in generations [42] shows that gradual knowledge transfer is more effective. We note that teachers should guide students step by step. Our approach is to gradually enhance the guidance role of the teacher in each distillation cycle, which has been experimentally proven to be simple and effective. Gradually reducing the proportion of standard cross entropy during the distillation process is conducive to the smooth transition of the loss function, preventing large changes in the gradient value from causing the network to collapse. At the same time, gradually increasing Kullback–Leibler divergence loss and strengthening the guidance role of the teacher are conducive to students' better learning.

To optimize model form θ_t to θ_{t+1} at the t-th iteration in distillation process, operations are a similar form of independent training.

$$\theta_{t+1} = \theta_t - \eta * \frac{\partial \mathcal{L}_i}{\partial \theta_t} \tag{12}$$

where η is the learning rate, (x_t, y_t) is the input data of the i-th distillation iteration, and $\frac{\partial \mathcal{L}_i}{\partial \theta_t}$ is the partial derivative gradient of distillation loss $\mathcal{L}_i$ to θ_t.

The student model performs $\mathbb{T}_d$ epoch distillation training in a training cycle. After the student model has trained the model through $\mathbb{T}_{in}$ rounds of independent training and $\mathbb{T}_d$ rounds of distillation training in a cycle, the model continues as the starting point for next cycle.

The deployment and training processes of the model are shown in Algorithm 1. Unlike general distillation methods, independent training and distillation training are included in one training cycle, while the aggregated information representing the teacher is updated in each new cycle. Student models participating in the training can have different network structures, but all students follow the same training process. End training until all student models are in a state of convergence. It is not allowed to exit early, so as to avoid a reduction in the overall amount of information in the

model. After training is completed, select the best model among students for deployment. A collection of multiple student networks may also be chosen under conditions of sufficient memory resources and computational resources.

Algorithm 1: Training process for student F_s

Input: Training data $\mathbb{X}$, corresponding label $\mathbb{Y}$, independent training epoch number $\mathbb{T}_{in}$, distillation training epoch number $\mathbb{T}_d$,temperature parameter t.
Output: Target model Θ_i of training completion
Random initialize Θ_i;
while *Not all student converged* **do**
  set a=0,r=0;
  while a<$\mathbb{T}_{in}$ **do**
    Calculating standard class probability output p_i^k (Equation (2));
    Calculate cross-entropy loss (Equation (1));
    SGD backpropagation optimization model Θ_i (Equation (3));
    Testing;
  Sending soft labels $\widetilde{S}_i$ to server;
  Receiving aggregated soft labels $\widehat{S}_e$ from server;
  while r<$\mathbb{T}_d$ **do**
    Calculating standard class probability p_i^k and soft probability $\widetilde{p}_i^k$ (Equations (2, 4));
    Calculate cross entropy loss and Kullback–Leibler divergence (Equations (1, 9));
    Get ultimate loss $\mathcal{L}_i$ (Equations (10, 11));
    SGD backpropagation optimization model Θ_i (Equation (12));
    Testing;
Model deployment: Θ_i;

4. Experiments

4.1. Experiment Settings

Datasets and Evaluation: We used three widely multiclass classification benchmarks: CIFAR10, CIFAR100 [43], and ImageNet [12]. For performance metrics, we adopted the common top-n $(n = 1)$ classification accuracy.

Neural Networks: We used seven networks in our experiments, AlexNet [44], VGG19 [45], ResNet18, ResNet50, ResNet110 [3], SqueezeNet1-1 [18], and DenseNet100 [46].

Implementation Details: We used the PyTorch framework and python gRPC (communication uses) to conduct all the following experiments. The student in the experiment was configured with an Nvidia 1080Ti graphics card, and the server could use the host of the CPU processor. We followed the training strategy and first initialized learning rate 0.1, then dropped the rate from 0.1 to 0.01 halfway (50%) through training, and to 0.001 at 75%. For the hyperparameters involved in the experiment, we used $\mathbb{T}_{in} = 10, \mathbb{T}_d = 30$ in the CIFAR experiment, while $\mathbb{T}_{in} = 5, \mathbb{T}_d = 10$ in ImageNet.

4.2. Comparison with Vanilla Independent Learning

Experiment Results on CIFAR Table 1 compares the top-1 accuracy performance of varying-capacity state-of-the-art network models trained by independent conventional training and our collaborative online-distillation learning algorithms on CIFAR10/CIFAR100. From Table 1, we can make the following observations: (1) All the different networks benefit from our collaborative online-distillation learning algorithm, particularly when small models collaboratively learn with large-capacity models. Top-1 accuracy improved by 5.94% (49.79–43.85) for AlexNet when training together with VGG. This suggests a generic superiority of our online knowledge distillation across

different architectures. (2) Performance gains on classification tasks with more classes (CIFAR100 VS CIFAR10) were higher for all networks. This is reasonable because richer interclass knowledge is transferred across individual architectures in an online manner to facilitate model optimization, indicating the favorable scalability of our method in solving large classification problems. (3) All of the large-capacity models benefit from joint training with small networks. Especially when ResNet collaboratively trained with VGG, top-1 accuracy improved by 3.73% (77.75–74.02).

Table 1. Top-1 accuracy (%) on the CIFAR10 and CIFAR100 datasets. I represents independent training and MD represents our method. m1 and m2 are the abbreviations of Model1 and Model2.

CIFAR10 Results					
Model1	**Model2**	**I (m1)**	**I (m2)**	**MD (m1)**	**MD (m2)**
VGG	ResNet	93.36	93.84	**94.27**	**95.6**
DenseNet	ResNet	95.35	93.84	**95.72**	**95.48**
AlexNet	VGG	77.58	93.36	**80.23**	**93.81**
ResNet	ResNet	93.84	93.84	**95.72**	**95.75**
DenseNet	DenseNet	95.35	95.35	**95.62**	**95.76**
VGG	VGG	93.36	93.36	**94.10**	**94.33**
CIFAR100 Results					
Model1	**Model2**	**I (m1)**	**I (m2)**	**MD (m1)**	**MD (m2)**
VGG	ResNet	72.57	74.02	**74.85**	**77.75**
DenseNet	ResNet	77.57	74.02	**78.57**	**79.09**
AlexNet	VGG	43.85	72.57	**49.79**	**74.08**
ResNet	ResNet	74.02	74.02	**77.81**	**77.47**
DenseNet	DenseNet	77.57	77.57	**78.55**	**78.85**
VGG	VGG	72.57	72.57	**75.43**	**75.45**

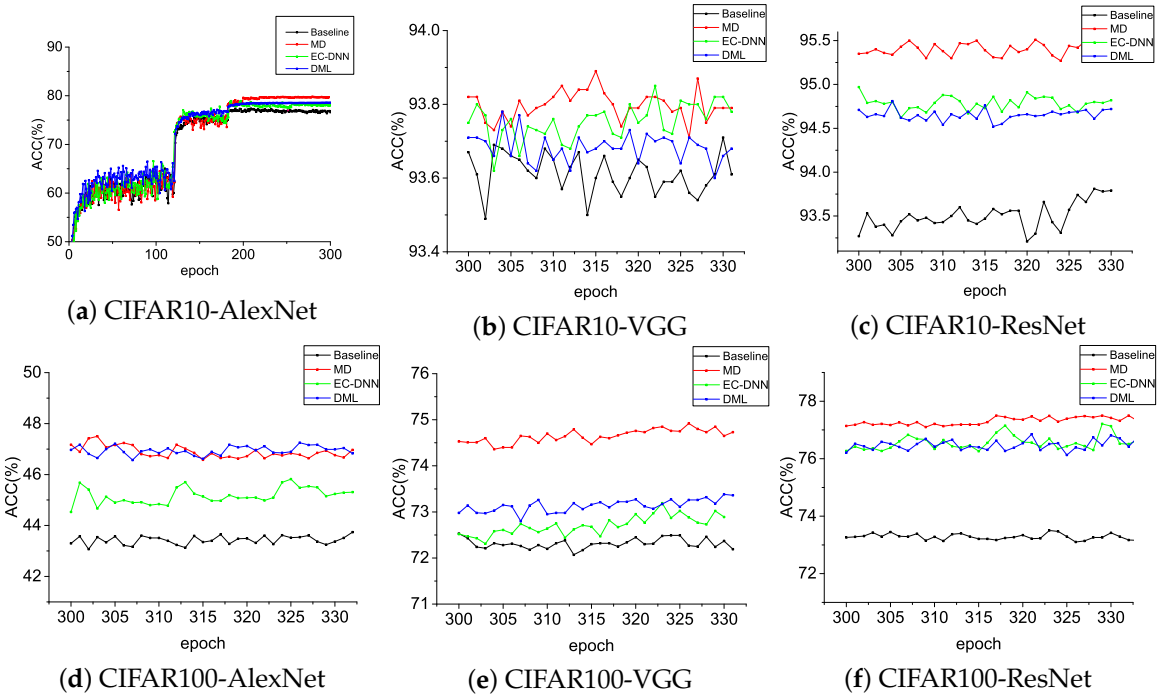
Experiment Results on ImageNet We tested the large-scale ImageNet with three different networks: ResNet-18, ResNer-50, and Squeeze1-1; results are shown in Table 2. Overall, we observed a similar performance comparison with these networks as on CIFAR10/CIFAR100. This indicates the superiority of our method when testing on large-scale image-classification settings. ImageNet is large, containing more than 1000 types of image datasets. On ImageNet, we verified that this method is effective for complex tasks with very large datasets.

Table 2. Experiment test accuracy (%) on ImageNet.

Model1	**Model2**	**I (m1)**	**I (m2)**	**MD (m1)**	**MD (m2)**
ResNet18	ResNet18	69.76	69.76	**70.53**	**70.44**
ResNet18	ResNet50	69.76	76.15	**70.65**	**76.38**
ResNet18	SqueezNet	69.76	58.1	**70.11**	**58.98**

4.3. Comparison with Conventional Distillation Methods

DML and Ensemble-Compression (EC-DNN) are the two most advanced distillation methods. In DML, we use two models for mutual distilling, while the EC-DNN method uses ensemble compression to improve model performance. This shows that our operation is effective: independent training was added to the process of model distillation, which increased the model difference with gradually incremental distillation weights to fuse polymerization information in distillation. Our approach still has tremendous advantages compared with state-of-the-art distillation methods, DML [25] and EC-DNN [47]. Table 3 shows that our method, MD, is almost always higher than DML and EC-DNN on CIFAR10/CIFAR100 on three typical networks: AlexNet, VGG, and ResNet. In Figure 4, wee see that MD continuously achieved higher Top-1 accuracy.



(**a**) CIFAR10-AlexNet (**b**) CIFAR10-VGG (**c**) CIFAR10-ResNet

(**d**) CIFAR100-AlexNet (**e**) CIFAR100-VGG (**f**) CIFAR100-ResNet

Figure 4. Classification accuracy compared to state-of-the-art online distillation. Baseline is the node-training-alone method; EC-DNN, Ensemble-Compression method; DML, Deep Mutual Learning method; MD, our method.

Table 3. Comparison with state-of-the-art online-distillation methods. **Red**/**Blue**: Best and second-best results.

Network	**AlexNet**		**ResNet-110**	
Datasets	**CIFAR10**	**CIFAR100**	**CIFAR10**	**CIFAR100**
Baseline	77.58	43.85	93.84	74.02
DML	78.6	**47.32**	94.81	76.92
EC-DNN	**78.67**	46.08	**94.97**	**77.38**
MD	**79.87**	**50.13**	**95.56**	**77.81**

4.4. Ablation Study

Effect on Student Number. Cluster learning enhances the effects of model aggregation. Intuitively, different students have individual prediction distributions on the same samples. In the process of aggregating the distribution of students, the more students involved, the smaller the common error. The experiment results of AlexNet and VGG19 on CIFAR10 in Figure 5 show that the more nodes that participate in training, the better the training effect is. However, we also noticed that the bigger the network number is, the slower the growth of model accuracy.

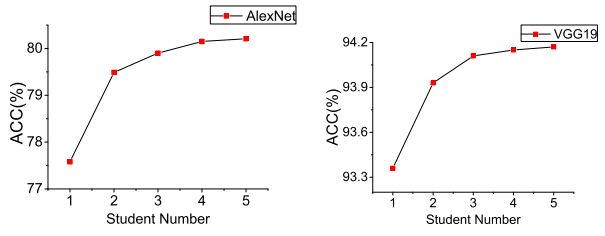


Figure 5. Effect on student number on CIFAR10.

Analysis of Experimental Parameters. We explored the effects of important parameters in our method, including temperature parameter t, the number of independent training epochs $\mathbb{T}_{in}$ and distillation epochs $\mathbb{T}_d$. On the CIFAR10 data set, we use two nodes for cooperatively distillation and setting $\mathbb{T}_{in} = 10, \mathbb{T}_d = 30$ to explore the influence of temperature parameters t. The results are shown in Table 4. We have the following conclusions:

1. The temperature parameters t have less impact for the same structures (AlexNet + AlexNet) compared to different structures (AlexNet + VGG19). We compared the logits statistics of AlexNet and VGG19 after 50 epochs training on CIFAR10. For AlexNet, the maximum, minimum and variance of logits are (24.41, -24.2, 8.15) while (20.5, -11.3, 11.42) for VGG19. It is more important to use an appropriate temperature parameter to unify the classification score scale for networks with different structures.

2. In the experiments of (AlexNet + VGG19), the best results (AlexNet: 80.23, VGG: 93.81) are obtained with $t = 3$. The classification accuracy decreases when $t > 5$, which show that a bigger t obscures the major class probability. The gradient explosion occurs when $t = 0.5$ or 1. A smaller t leading to the loss of secondary class information, and the values of soft labels are close to discrete 0 and 1 which causes the difficulty in model optimization with Kullback-Leibler divergence.

Table 4. Setting the number of independent training epochs $\mathbb{T}_{in} = 10$ and the number of distillation epochs $\mathbb{T}_d = 30$, the influence of temperature parameters on accuracy (%) is studied. The same structure (AlexNet + AlexNet) and different structure (AlexNet + VGG19) are compared in the experiment.

Temperature (t)	**0.5**	**1**	**2**	**3**	**4**	**5**	**6**	**9**
AlexNet	NaN	79.37	79.3	**79.51**	79.3	79.43	79.35	79.42
AlexNet	NaN	79.56	79.21	79.59	79.46	**79.68**	79.28	79.48
AlexNet	NaN	NaN	79.97	80.23	80.22	**80.44**	79.79	80.17
VGG19	NaN	NaN	93.72	**93.81**	93.41	93.46	93.67	93.68

Setting temperature parameter $t = 3$, the relationship between the number of independent training epochs $\mathbb{T}_{in}$ and distillation training epochs $\mathbb{T}_d$ is shown in Table 5:

Table 5. Setting temperature parameter $t = 3$, the number of independent training epochs $\mathbb{T}_{in} = 10$, change the number of distillation epochs $\mathbb{T}_d$, the accuracy (%) of cooperative training of two nodes (AlexNet + AlexNet) using our method.

Distillation Epochs ($\mathbb{T}_d$)	**1**	**5**	**10**	**20**	**30**	**40**
AlexNet	77.72	78.29	78.9	78.84	**79.43**	79.41
AlexNet	77.96	78.54	78.96	79.04	**79.59**	79.31

We observed that the best results were obtained when $\mathbb{T}_{in} = 10, Td = 30$. Fixed $\mathbb{T}_{in} = 10$, with the increase of the number of distillation epochs $\mathbb{T}_d$, the accuracy increases gradually and stop increasing after $\mathbb{T}_d = 30$. Such changes show that a short distillation process cannot adequately transfer teacher's knowledge to the student networks, resulting in the loss of information.

Gradual Knowledge Transfer. An article on knowledge distillation in generations [42] suggests that teachers who are tolerant usually educate better teachers and produce better results by gradually increasing the constraints of the teacher. We used a multicycle training process in which gradual strict KL constraints are used in each cycle of the distillation phase. Validated on CIFAR10 and CIFAR100, our method compares the use of generally incremental distillation weights ((gMD), $\mathbb{T}_{in} = 10, \mathbb{T}_d = 30$, $\alpha = 1 - r/\mathbb{T}_d, \beta = 1 + r/\mathbb{T}_d$) with fixed distillation weights (fMD), $\mathbb{T}_{in} = 10, \mathbb{T}_d = 30, \alpha = 1, \beta = 2$). Experiment results are shown in Table 6.

Table 6. Results comparison of whether to use the gradual enhancement of the teacher-constraint method. gMD, incremental distillation weights; fMD, fixed distillation weights.

Dataset	CIFAR10			CIFAR100		
Model	**Baseline**	**fMD**	**gMD**	**Baseline**	**fMD**	**gMD**
AlexNet	77.58	79.49	79.87	43.85	49.96	50.13
VGG19	93.36	93.93	94.33	72.57	75.32	75.45

On both the CIFAR10 and CIFAR100 datasets, AlexNet and VGG19 were more accurate at using gMD than using fMD. Increasing teacher constraints enhances the distillation effect.

5. Conclusions and Future Work

In this paper, we proposed a collaborative framework using ensemble and distillation mechanisms, which helps participating nodes learn to integrate the network information of other nodes during training to achieve better performance. Our approach uses soft labels to pass model information and make it compatible with different networks and training methods. The use of progressively increasing distillation and adding independent-training phase constraints enhances the effectiveness of the method. The experiment verified that our method surpassed the best distillation method of the time. Our method has wide applicability in classification tasks. In the hopes of helping other researchers, competition members, and industries to train better networks, our future work will explore the method of intermediate distillation and the effectiveness of distillation in other scenarios.

Author Contributions: L.G. carried out the experiments and wrote the first draft of the manuscript. K.X. and X.L. conceived and supervised the study, and edited the manuscript. H.M. and D.F. contributed to the data analysis. All authors reviewed the manuscript.

Funding: This research was funded by the National Key R and D Program of China (2016YFB1000101).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436. [CrossRef] [PubMed]
2. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.
3. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 770–778.
4. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
5. Girshick, R. Fast r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
6. Li, H.; Li, Y.; Porikli, F. Deeptrack: Learning discriminative feature representations online for robust visual tracking. *IEEE Trans. Image Process.* **2016**, *25*, 1834–1848. [CrossRef] [PubMed]
7. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient estimation of word representations in vector space. *arXiv* **2013**, arXiv:1301.3781.
8. Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.S.; Dean, J. Distributed representations of words and phrases and their compositionality. In *Neural Information Processing Systems*; Neural Information Processing Systems Foundation, Inc.: Lake Tahoe, NV, USA, 2013; pp. 3111–3119.
9. Hermann, K.M.; Kocisky, T.; Grefenstette, E.; Espeholt, L.; Kay, W.; Suleyman, M.; Blunsom, P. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*; Palais des Congrès de Montréal: Montréal, QC, Canada, 2015; pp. 1693–1701.
10. Joulin, A.; Grave, E.; Bojanowski, P.; Mikolov, T. Bag of tricks for efficient text classification. *arXiv* **2017**, 427–431, arXiv:1607.01759.

11. Weston, J.; Bordes, A.; Chopra, S.; Rush, A.M.; van Merriënboer, B.; Joulin, A.; Mikolov, T. Towards AI-complete question answering: A set of prerequisite toy tasks. *arXiv* **2016**, arXiv:502.05698.
12. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*; Neural Information Processing Systems Foundation, Inc.: Lake Tahoe, NV, USA, 2012; pp. 1097–1105.
13. Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 3431–3440.
14. Zagoruyko, S.; Komodakis, N. Wide residual networks. *Br. Mach. Vis. Conf.* **2016**, *8*, 35–67.
15. Canziani, A.; Paszke, A.; Culurciello, E. An analysis of deep neural network models for practical applications. *arXiv* **2017**, arXiv:605.07678.
16. Deng, L.; Platt, J.C. Ensemble deep learning for speech recognition. In Proceedings of the Fifteenth Annual Conference of the International Speech Communication Association, Singapore, 14–18 September 2014.
17. Qiu, X.; Zhang, L.; Ren, Y.; Suganthan, P.N.; Amaratunga, G. Ensemble deep learning for regression and time series forecasting. In Proceedings of the IEEE Computational Intelligence in Ensemble Learning, Orlando, FL, USA, 9–12 December 2014; pp. 1–6.
18. Iandola, F.N.; Han, S.; Moskewicz, M.W.; Ashraf, K.; Dally, W.J.; Keutzer, K. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5 MB model size. *arXiv* **2017**, arXiv:1602.07360.
19. Cheng, Y.; Wang, D.; Zhou, P.; Zhang, T. A survey of model compression and acceleration for deep neural networks. *arXiv* **2017**, arXiv:1710.09282.
20. Han, S.; Mao, H.; Dally, W.J. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv* **2016**, arXiv:1510.00149.
21. Ba, J.; Caruana, R. Do deep nets really need to be deep? In *Advances in Neural Information Processing Systems*; Palais des Congrès de Montréal: Montréal, QC, Canada, 2014; pp. 2654–2662.
22. Hinton, G.; Vinyals, O.; Dean, J. Distilling the knowledge in a neural network. *arXiv* **2015**, arXiv:1503.02531.
23. Verma, N.; Mahajan, D.; Sellamanickam, S.; Nair, V. Learning hierarchical similarity metrics. In Proceedings of the IEEE Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 2280–2287.
24. Deng, J.; Berg, A.C.; Li, K.; Fei-Fei, L. What does classifying more than 10,000 image categories tell us? In *European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 71–84.
25. Zhang, Y.; Xiang, T.; Hospedales, T.M.; Lu, H. Deep mutual learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 4320–4328.
26. Dean, J.; Ghemawat, S. MapReduce: Simplified data processing on large clusters. *Commun. ACM* **2008**, *51*, 107–113. [CrossRef]
27. Dean, J.; Corrado, G.; Monga, R.; Chen, K.; Devin, M.; Mao, M.; Senior, A.; Tucker, P.; Yang, K.; Le, Q.V.; et al. Large scale distributed deep networks. In *Advances in Neural Information Processing Systems*; Neural Information Processing Systems Foundation, Inc.: Lake Tahoe, NV, USA, 2012; pp. 1223–1231.
28. Izmailov, P.; Podoprikhin, D.; Garipov, T.; Vetrov, D.P.; Wilson, A. Averaging weights leads to wider optima and better generalization. *arXiv* **2018**, arXiv:1803.05407, pp. 876–885.
29. Zhang, X.; Trmal, J.; Povey, D.; Khudanpur, S. Improving deep neural network acoustic models using generalized maxout networks. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, Florence, Italy, 4–9 May 2014; pp. 215–219.
30. Xu, K.; Mi, H.; Feng, D.; Wang, H.; Chen, C.; Zheng, Z.; Lan, X. Collaborative deep learning across multiple data centers. *arXiv* **2018**, arXiv:1810.06877.
31. Chen, J.; Pan, X.; Monga, R.; Bengio, S.; Jozefowicz, R. Revisiting distributed synchronous SGD. *arXiv* **2016**, arXiv:1604.00981.
32. Caruana, R. Multitask learning. *Mach. Learn.* **1997**, *28*, 41–75. [CrossRef]
33. Evgeniou, T.; Pontil, M. Regularized multi–task learning. In Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, WA, USA, 22–25 August 2004; pp. 109–117.
34. Sun, Y.; Chen, Y.; Wang, X.; Tang, X. Deep learning face representation by joint identification-verification. In *Advances in Neural Information Processing Systems*; Palais des Congrès de Montréal: Montréal, QC, Canada, 2014; pp. 1988–1996.

35. Yim, J.; Jung, H.; Yoo, B.; Choi, C.; Park, D.; Kim, J. Rotating your face using multi-task deep neural network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 676–684.
36. Safont, G.; Salazar, A.; Vergara, L. Multiclass alpha integration of scores from multiple classifiers. *Neural Comput.* **2019**, *31*, 806–825. [CrossRef] [PubMed]
37. Duan, Q.; Ajami, N.K.; Gao, X.; Sorooshian, S. Multi-model ensemble hydrologic prediction using Bayesian model averaging. *Water Resour.* **2007**, *30*, 1371–1386. [CrossRef]
38. Soriano, A.; Vergara, L.; Ahmed, B.; Salazar, A. Fusion of scores in a detection context based on alpha integration. *Neural Comput.* **2015**, *27*, 1983–2010. [CrossRef] [PubMed]
39. Anil, R.; Pereyra, G.; Passos, A.T.; Ormandi, R.; Dahl, G.E.; Hinton, G.E. Large scale distributed neural network training through online distillation. *arXiv* **2018**, arXiv:1804.03235.
40. Tarvainen, A.; Valpola, H. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems*; Neural Information Processing Systems Foundation, Inc.: Long Beach, CA, USA, 2017; pp. 1195–1204.
41. Lan, X.; Zhu, X.; Gong, S. Knowledge distillation by On-the-Fly native ensemble. *arXiv* **2018**, 7528–7538, arXiv:1806.04606.
42. Yang, C.; Xie, L.; Qiao, S.; Yuille, A.L. Knowledge distillation in generations: More tolerant teachers educate better students. *arXiv* **2018**, arXiv:1805.05551.
43. Krizhevsky, A.; Hinton, G. *Learning Multiple Layers of Features from Tiny Images*; Technical Report; University of Toronto: Toronto, ON, Canada, 2009.
44. Krizhevsky, A. One weird trick for parallelizing convolutional neural networks. *arXiv* **2014**, arXiv:1404.5997.
45. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2015**, arXiv:1409.1556.
46. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; p. 3.
47. Sun, S.; Chen, W.; Bian, J.; Liu, X.; Liu, T.Y. Ensemble-compression: A new method for parallel training of deep neural networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*; Springer: Cham, Switzerland, 2017; pp. 187–202.