Hindawi
Complexity
Volume 2022, Article ID 5578043, 13 pages
https://doi.org/10.1155/2022/5578043

*Research Article*

# The Classification of Multi-Domain Samples Based on the Cooperation of Multiple Models

**Qingzeng Song**,[1] **Junting Xu**,[1] **Lei Ma**,[2] **Ping Yang**,[1] **and Guanghao Jin**[2]

[1]*School of Computer Science and Technology, Tiangong University, Tianjin 300387, China*
[2]*School of Telecommunication Engineering, Beijing Polytechnic, Beijing 100176, China*

Correspondence should be addressed to Guanghao Jin; jingh_research@163.com

Received 5 May 2022; Revised 8 August 2022; Accepted 18 August 2022; Published 25 September 2022

Academic Editor: Dimitri Volchenkov



This article proposed a novel classification framework that can classify the samples of multiple domains based on the outputs of multiple models. Different from the existing methods that train single model on all domains, our framework trains multiple models on each domain. On a testing sample, the outputs of all trained models are used to predict the domain of this sample. Then, this sample is classified by the output of models that belong to the predicted domain. Experiments show that our framework achieved higher accuracy than the existing methods. Furthermore, our framework achieves good scalability on multiple domains.

## 1. Introduction

In these days, deep learning models can achieve good performance in many applications [1–5]. Generally, the performance of a deep learning model depends on the captured features [6–10]. More domains are important to the scalability of deep learning system while these increase the difficulty of training high-performance models. Figure 1 introduces an example that includes multiple domains. We use the dataset to present the collected samples of the corresponding domain. These datasets may have a different number of labels and various samples, which increase the difficulty of training high-performance models. Transfer learning [11–13] can temporarily solve this problem while the performance is still limited by the structure of models. As Figure 1(a) introduces, a bigger model (deeper structure and more layers) is a general solution as this can capture more features while this is limited by the computational resource or may cause the vanishing gradient problem [14–16]. Thus, there should be another way to improve the performance of deep learning system on multiple domains like using multiple models.

Some fusion methods [17–19] can utilize multiple models to increase the classification accuracy, where performance depends on the selection of high accurate trained models. Compared with training a single model on multiple domains, training each model on the corresponding domain can be a good solution as this can achieve good scalability as we introduced in Figure 1(b). Generally, the trained model easily ensures high accuracy in the corresponding domain. On the contrary, this model may have low accuracy on the other domains, which reduces the performance of these fusion methods. Thus, on a testing sample, the prediction of the domain is important to the fusion methods.

In this article, we built a novel framework (CMS-CMM, the Classification of Multi-Domain Samples Based on the Cooperation of Multiple Models) to increase the accuracy of classification on the samples of multiple domains. Our contribution can be summarized as the following. (1) We built a novel framework that achieves the scalability of the deep learning system. As the number of domains increased, the difficulty of transfer learning is increased as it has to consider the performance of all domains. On the contrary, our framework only needs to train some deep learning models on the training set of a new domain, which benefits the scalability. (2) Our framework increases the accuracy of classification without increasing the structure of models. Generally, a bigger model increases the classification accuracy while it needs more space of memory that is impossible to satisfy in some applications. Instead, our

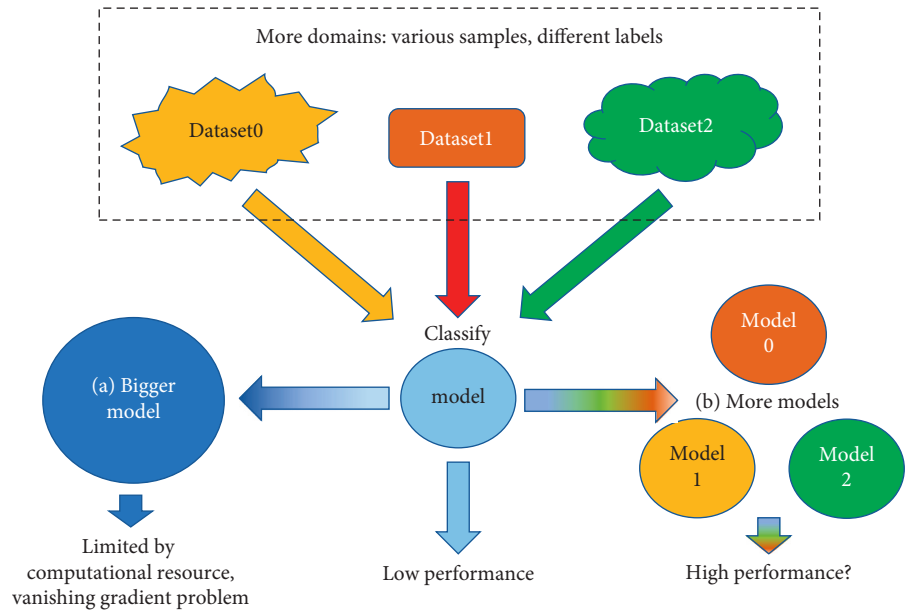


Figure 1: The solutions to classify the samples of multiple domains. (a) Bigger model. (b) More models.

framework only increases the number of models where each of them lowers the consumption of memory space compared with that of a big model.

The rest of the article is organized as follows: Section 1.1 introduces the existing methods and their problems. In Section 2, we present our framework and related analyses. The experiment is organized in Section 3. Section 5 gives the conclusion and future work.

*1.1. Related Works.* VoVNet-57 (Variety of View Network) is designed for object classification task, which consists of blocks including 3 convolution layers and 4 stages modules and outputs stride 32 [20]. The sample is passed through convolutional layers, where here the filters consist of a small receptive field. ResNeSt (residual networks) is a state-of-art deep learning model for image classification that uses a modular structure with a split-attention block and applies an attention mechanism to feature map groups [21]. From ResNeSt50 to ResNeSt269, the structure becomes bigger and more complicated, so that these can get higher accuracy especially when there are more and bigger size training samples. Based on the size of testing samples and computational resource, we use ResNeSt101 in this article. RepVGG (re-parameterization visual geometry group) is a classification model, which is improved on the basis of the existing models [22]. DenseNet (densely connected convolutional network) is a convolutional neural network with dense connections [23]. In this network, there is a direct connection between any two layers, which means the input of each layer connects to all the previous layers. VGG16 is a variant of VGG (visual geometry group) models for image classification [24]. ResNet (residual neural network) allows the original input information to be detoured directly to the output, which simplifies the process and reduces the difficulty of training [25].

Some fusion methods have been applied to improve the performance of classification, which applies multiple models [17]. In that article, weighted voting method achieved the highest accuracy among all of the other ones. Weighted voting is also utilized to construct a more reliable classification system [18]. A sliding window is applied to the weighted majority voting algorithm in that article. This method is applied to a DNN (deep neural network), a CNN (convolutional neural network), and an LSTM (long short-term memory) network to improve the performance [19]. These methods can combine the results of models to improve the accuracy. As the weights play an important role in the combination, there should be a validation set to compute these weights. Furthermore, more various models can benefit the improvement of the accuracy. In this article, we also apply these fusion methods to our system for higher accuracy with some optimizations.

When using these methods, the performance of each model is important. Training a model on the single domain can ensure high accuracy on this domain while it may cause low accuracy on the other domains. At the same time, training a model on multiple domains may reduce the accuracy on each domain. Thus, our framework tries to solve this problem, which is introduced in the next section.

## 2. Our Framework

Before giving the details of our framework, we give the following definitions. These definitions are to explain the implementation of the methods.

*2.1. Preliminaries.* We set $S_n$ as a sample and $L_k$ as the label of an object. We set $L_{\text{groundtruth}}$ as the ground truth on $S_n$ where $L_{\text{groundtruth}} \in \{L_k\}$ [26, 27]. The label is to benefit the computation, which is generally a number [28, 29]. For

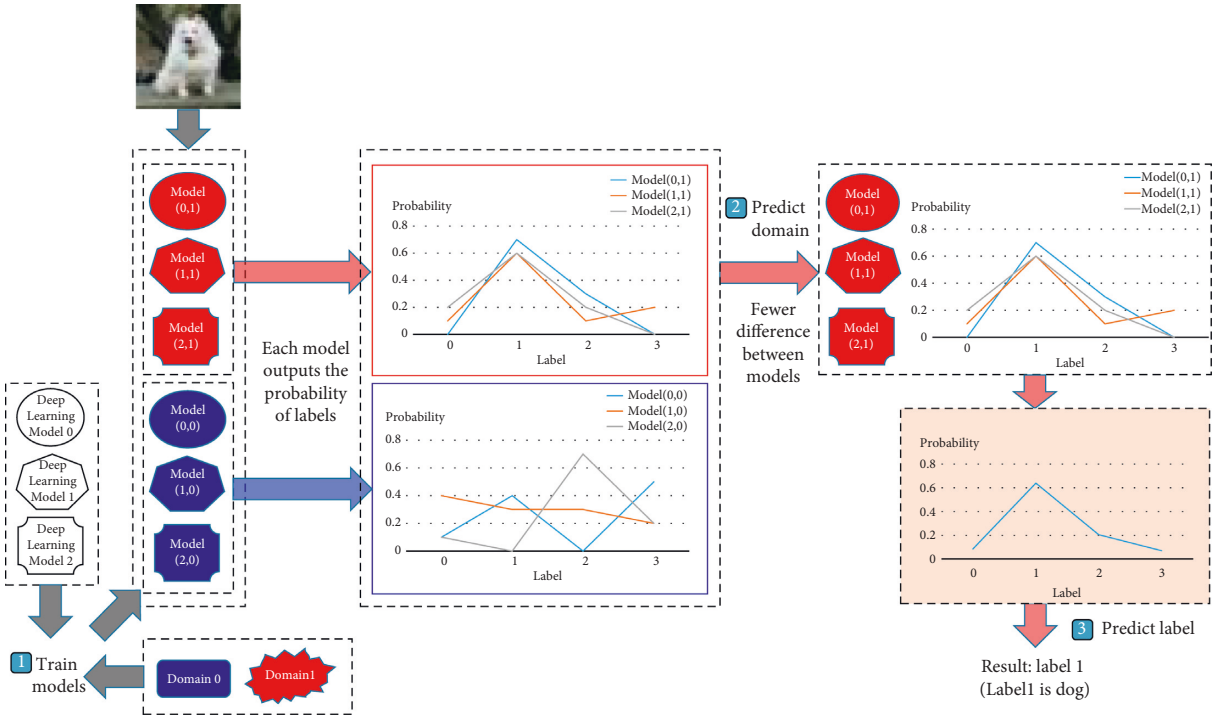


Figure 2: The illustration of the framework of our methods.

example, when there are 10 objects to be classified, the label is from 0 to 9.

*2.2. The Illustration of Our Framework.* Figure 2 illustrates our framework, which is named as *CMS-CMM*. At the first step, our framework trains some existing deep learning models (like deep learning models 0, 1, and 2 in this figure) on each domain (like domains 0 and 1 in this figure). Then, on a testing sample, each model outputs the probability of labels. Firstly, based on the difference in these probabilities, we can predict the domain of this sample (illustrated by the chart). Secondly, we select the trained models of the predicted domain. Then, we can use the output of these models to predict the label of this sample (illustrated by the chart).

*2.3. Training the Models and Outputting the Probability of Labels.* We select a deep learning model $M_i$. Then, we train $M_i$ on a domain $D_u$ to get a trained model $M_{i,u}$. We define $P(M_{i,u}(S_n) = L_k)$ as the probability of label $L_k$ on the sample $S_n$, which is the output by the trained model $M_{i,u}$. Generally, the most possible result is selected by the following equation:

$$L_{\text{result}} = \text{argmax}_{L_k} P\left(M_{i,u}(S_n) = L_k\right). \tag{1}$$

Which is used as the predicted result.

*2.4. Predicting the Domain.* In our framework, we firstly select some existing deep learning models $\{M_i\}$. Then, we train these models on each domain $D_u$ to get a set of trained models $\{M_{i,u}\}$. When we assume a sample $S_n$ belongs to a domain $D_u$, we can get a probability of labels $\{P(M_{i,u}(S_n) = L_k)\}$ by a model $M_{i,u}$. Then, by the other model $M_{j,u}$, we can also get $\{P(M_{j,u}(S_n) = L_k)\}$. We define the difference between the model $M_{i,u}$ and the model $M_{j,u}$ on a sample $S_n$ as follows:

$$\left\|M_{i,u}(S_n) - M_{j,u}(S_n)\right\|_{\{L_k\}} = \sum_{L_k} \left|P\left(M_{i,u}(S_n) = L_k\right) - P\left(M_{j,u}(S_n) = L_k\right)\right|. \tag{2}$$

We can define the difference between $M_{i,u}$ and $\{M_{j,u}\}$ on a sample $S_n$ as follows:

$$\sum_j \|M_{i,u}(S_n) - M_{j,u}(S_n)\|_{\{L_k\}} = \sum_j \sum_{L_k} \left|P\left(M_{i,u}(S_n) = L_k\right) - P\left(M_{j,u}(S_n) = L_k\right)\right|. \tag{3}$$

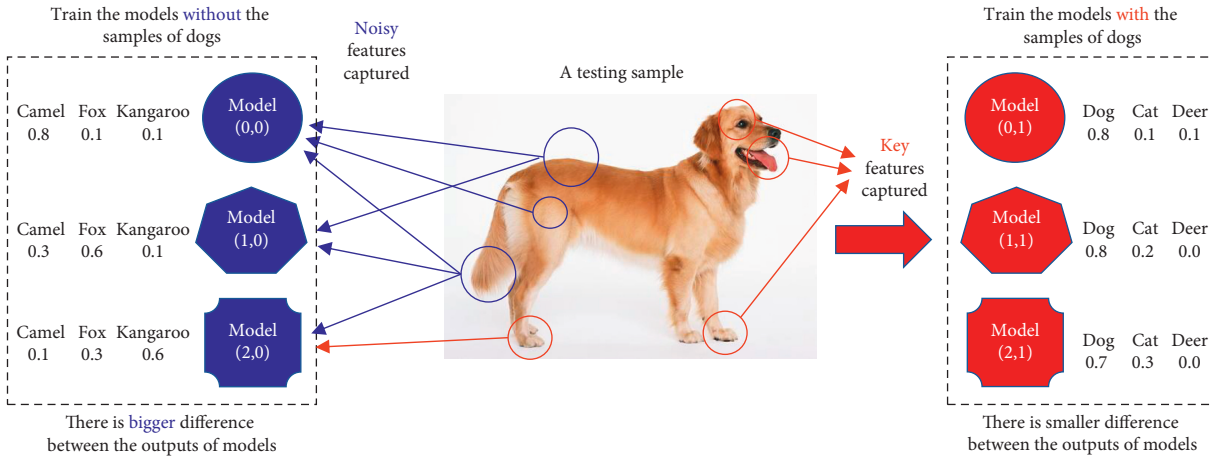


Figure 3: The output of trained models with and without dog.

Generally, we can select $M_{i,u}$ as the model that can achieve the highest accuracy on the validation set. Thus, (3) is to present the difference between the highest accurate model with the other ones. Then, we can select the domain $D_{\text{result}}$ as the predicted domain of sample $S_n$ as below:

$$D_{\text{result}} = \text{argmin}_{D_u} \sum_j \|M_{i,u}(S_n) - M_{j,u}(S_n)\|_{\{L_k\}}. \tag{4}$$

Figure 3 uses an example to explain how to predict the domain. We assume that the domain $D_1$ contains the sample and there are three models trained on this domain, which are $M_{0,1}, M_{1,1}, M_{2,1}$. Thus, these models can well capture the key features of this sample, which leads the probability of dog being high and those of other labels being low. We assume the domain $D_0$ does not contain the samples of dog. Then, we can also get corresponding three trained models $M_{0,0}$, $M_{1,0}$, $M_{2,0}$ of this domain. As these models have not captured the features of dog on the training set, these models may capture noisy features of dog (also included by other labels), which causes big difference between the outputs of these models.

*2.5. Predicting the Label.* Once our framework has predicted the domain of a sample, we can use the corresponding models that are trained on this domain to predict the label of this sample. To increase the accuracy of the prediction, our framework uses the fusion method [19] that is weighted model average as follows:

$$W_i = \frac{1}{\left(1 - P\left(M_{i,u}(S_n) = L_{\text{goundtruth}}\right)\right)}, W_i = \frac{W_i}{\sum_j W_j},$$
$$L_{\text{result}} = \text{argmax}_{L_k} \sum_i P\left(M_{i,u}(S_n) = L_k\right) \times W_i, \tag{5}$$

where $W_i$ presents the weight that is applied to the output of models. By using these weights, the output of higher accurate model plays a more important role to the final result. We can compute the $W_i$ by using the validation set. We name our framework with this optimization as *CMS-CMM* from now on.

*2.6. Optimization by the Distribution of the Labels.* There are two cases that may cause the wrong prediction of the domain. Figure 4 introduces the two cases. We assume that domain 0 has 100 labels and domain 1 has 10 labels. By the general setting, the trained models of domain 0 will output the probability of 100 labels. At the same time, the trained models of domain 1 will output the probability of 10 labels. In 4(a) of this figure, we input the testing sample of domain 0 into the trained models. The difference between the trained models of domain 1 may be lower than that between the trained models of domain 0 occasionally because the range of error labels is reduced. Especially when the accuracy of models is low, this case easily causes wrong prediction of the domain. In 4(b) of this figure, we input the testing sample of domain 1 into the trained models. The difference between the trained models of domain 0 may be lower than that between the trained models of domain 1 occasionally as the error labels are scattered to a wider range. Especially when the accuracy of models is high, this case also easily causes the wrong prediction of domain.

To solve this problem, we make all of the trained models predict the same number of labels. For example, the trained models on $D_0$ (100 labels) or $D_1$ (10 labels) can predict 10 labels, which is the maximum number of labels among these domains. Then, when the testing samples belong to $D_1$, we only consider labels 0 to 9 as the possible correct one. Thus, we revise equation (4) as the following when there are different number of labels between domains.

$$D_{\text{result}} = \text{argmin}_{D_u}\left(\sum_j \|M_{i,u}(S_n) - M_{j,u}(S_n)\|_{\{\overline{L}_k\}} + \varphi_u \times \sum_j \|M_{i,u}(S_n) - M_{j,u}(S_n)\|_{\{\overline{L}_k\}}\right), \tag{6}$$

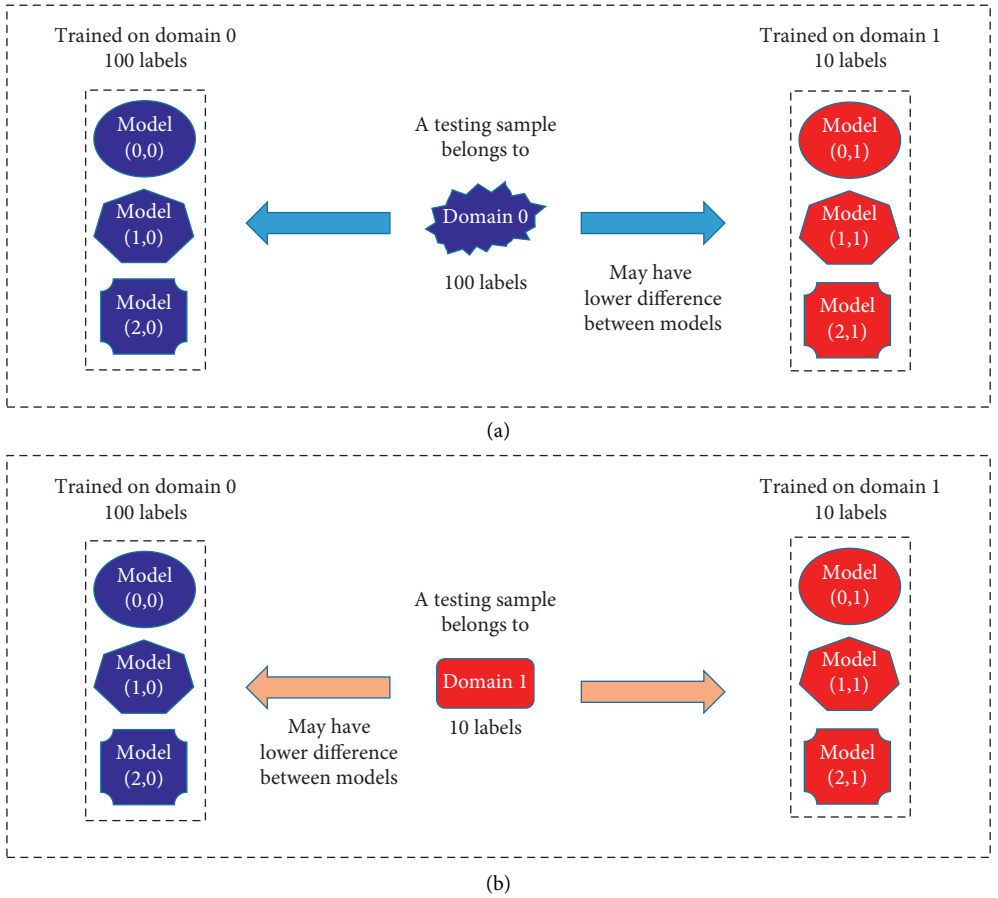


Figure 4: The two cases causing the wrong prediction of domain.

where $\{L_k\}$ is the possible correct labels of the corresponding domains $D_u$. We set $\{\overline{L}_k\}$ as the remain labels. For example, when the $D_0$ contains 100 labels and the domain $D_1$ contains 10 labels, we can set $\overline{L}_k$ is from 10 to 99 for $D_1$. Thus, all models of these domains output the probability of the same number labels. We can use $\varphi_u$ by using the validation set. We name our framework with this optimization as *CMS-CMM-opt* from now on.

## 3. Experiment

We evaluate our methods with the existing ones on some real datasets. When we randomized the parameters, we evaluate 1000 times. We trained the deep learning models (VoVNet-57 [20], ResNeSt50 [21], RepVGG [22], DenseNet [23], VGG16 [24], and ResNet [25]) on some real datasets by the reported default settings of these models. We set the number of epochs [30, 31] as 10 for all these models on any training set. We do not focus on the designing of structure or tuning the hyper-parameters. Instead, we focus on how to use multiple models to achieve the scalability and ensure high accuracy at the same time. We set a random number of validation samples, which is from 500 to 800.

*3.1. Introduction of the Datasets.* CIFAR-10 [32, 33] has 50000 training samples and 10000 testing samples that belong to 10 labels. We use 50000 training samples to train the models. Then we have 10000 samples left to the validation and testing. CIFAR-100 dataset is just like the CIFAR-10, except it has 100 classes containing 600 images each [34, 35]. There are 500 training images and 100 testing images per label. We use 50000 training samples to train the models. Then we have 10000 samples left to the validation and testing. The Mini-ImageNet [36, 37] dataset is for few-shot learning evaluation. Its complexity is high due to the use of ImageNet images but requires fewer resources and infrastructure than running on the full ImageNet dataset. We use 48000 training samples to train the models. Then we have 12000 samples left to the validation and testing. EuroSAT [38, 39] dataset is based on satellite images consisting of 10 classes with 27000 labelled samples. We use 21600 as training samples and 5400 as the testing ones. Intel Image Classification [40] dataset contains natural scenes around the world. There are around 14 k images for training, 3 k for testing, and 7 k in prediction (without labels).

*3.2. Introduction of the Evaluation Metrics.* We introduce some metrics to compare the methods in different dimensions on the testing samples. We assume a sample $S_n$ belongs to the domain $D_{\text{groundtruth}}$ and the corresponding ground truth of label is $L_{\text{groundtruth}}$. We define $L_{\text{result}}$ is the predicted result of label and $D_{\text{result}}$ as the predicted result of domain by a method. Then we can define the following evaluation metrics.

Table 1: Evaluation based on CD (the accuracy of predicting correct domains).

| Methods | CIFAR-10 (%) | CIFAR-100 (%) | Mini-ImageNet (%) | EuroSAT (%) | Intel image classification (%) |
|---|---|---|---|---|---|
| Maximum appeared method | 79.21 | 40.93 | 50.38 | 55.04 | 53.26 |
| Fusion method [19] | 71.79 | 40.32 | 61.46 | 90.91 | 83.03 |
| CMS-CMM | 85.59 | 71.27 | 73.65 | 92.48 | 65.70 |
| CMS-CMM-opt | 95.78 | 77.63 | 75.52 | 99.09 | 90.62 |

*CD* presents the accuracy of predicting correct domains as below:

$$\mathrm{CD} = \frac{\sum_{D_{\mathrm{result}}=D_{\mathrm{groundtruth}}} \sum_{L_{\mathrm{result}}} 1}{\left(\sum_{D_{\mathrm{result}}} \sum_{L_{\mathrm{result}}} 1\right)}, \tag{7}$$

where the higher one is better. *CDCL* presents the accuracy of predicting correct domains and correct labels as below:

$$\mathrm{CDCL} = \frac{\left(\sum_{D_{\mathrm{result}}=D_{\mathrm{groundtruth}}} \sum_{L_{\mathrm{result}}=L_{\mathrm{groundtruth}}} 1\right)}{\left(\sum_{D_{\mathrm{result}}} \sum_{L_{\mathrm{result}}} 1\right)}, \tag{8}$$

where the higher one is better. *CDWL* presents that the percentage of predicting correct domains and wrong labels as below:

$$\mathrm{CDWL} = \frac{\left(\sum_{D_{\mathrm{result}}=D_{\mathrm{groundtruth}}} \sum_{L_{\mathrm{result}}, L_{\mathrm{groundtruth}}} 1\right)}{\left(\sum_{D_{\mathrm{result}}} \sum_{L_{\mathrm{result}}} 1\right)}, \tag{9}$$

where the lower one is better.

*WD* presents the percentage of predicting wrong domains as below:

$$\mathrm{WD} = \frac{\left(\sum_{D_{\mathrm{result}} \neq D_{\mathrm{groundtruth}}} \sum_{L_{\mathrm{result}}} 1\right)}{\left(\sum_{D_{\mathrm{result}}} \sum_{L_{\mathrm{result}}} 1\right)}, \tag{10}$$

where the lower one is better. *WDCL* presents the percentage of predicting wrong domains and correct labels as below:

$$\mathrm{WDCL} = \frac{\left(\sum_{D_{\mathrm{result}} \neq D_{\mathrm{groundtruth}}} \sum_{L_{\mathrm{result}}=L_{\mathrm{groundtruth}}} 1\right)}{\left(\sum_{D_{\mathrm{result}}} \sum_{L_{\mathrm{result}}} 1\right)}, \tag{11}$$

where the lower one is better. When the prediction of the domain goes wrong, the predicting of labels is meaningless as the labels of different datasets indicates different kinds of objects. *WDWL* presents the percentage of predicting wrong domains and wrong labels as below:

$$\mathrm{WDWL} = \frac{\left(\sum_{D_{\mathrm{result}} \neq D_{\mathrm{groundtruth}}} \sum_{L_{\mathrm{result}} \neq L_{\mathrm{groundtruth}}} 1\right)}{\sum_{D_{\mathrm{result}}} \sum_{L_{\mathrm{result}}} 1}, \tag{12}$$

where the lower one is better.

*3.3. Evaluation of Domain Prediction.* We do not use additional information (like the resolution or size of sample) to predict the correct domain of samples. In Table 1, Maximum appeared method predicts the domain by the maximum appeared label. In more detail, we select the result that appeared maximum times from the trained models of each domain. Among all these results, we select the one that appeared maximum times and set the corresponding domain as the predicted domain. Following the same way, the fusion method [19] predicts the domain by the maximum value of weight probabilities.

We used CD (the accuracy of predicting correct domains) to evaluate the methods. As we can see in Table 1, our CMS-CMM-opt achieves higher accuracy than the existing methods, which is 16.62% higher on average. Furthermore, CMS-CMM-opt achieves higher accuracy than our CMS-CMM, which proves the efficiency of the optimization.

*3.4. Evaluation of Label Classification.* Our final objective is to classify the samples. Thus, based on the prediction of domain, there must be also the classification of the samples at the following step. Thus, on a testing sample, only when a method correctly predicted the domain and label at the same time, we admit this method correctly output the result. For example, the label 9 of CIFAR-10 and the label 9 of CIFAR-100 mean different kinds of objects.

We used CDCL (the accuracy of predicting correct domains and correct labels) to evaluate the methods. As we can see in Table 2, our CMS-CMM-opt achieves higher accuracy than the existing methods, which is 14.01% higher on average. Compared with the domain prediction, the increase of the accuracy is reduced from 16.62% to 14.01% because there is also error when predicting the labels. CMS-CMM use the fusion method [19] as the prediction of labels after the domain prediction. We can see CMS-CMM-opt also achieves higher accuracy than CMS-CMM, which is 11.25% higher on average.

*3.5. Evaluation of the Scalability.* In this subsection, we do research about the scalability of our framework based on the metric of CDCL (the accuracy of predicting correct domains and correct labels). We added the domain one by one and computed the label classification as Table 3 shows.

As we can see in Table 3, the accuracy of CIFAR-10 remains the same as the number of domains becomes big. On the other side, the accuracies of CIFAR-100 and Mini-ImageNet becomes lower. The accuracy of each models plays important role to the classification accuracy. The other important factor to the accuracy is the similarity between domains, which will be introduced in the next subsection.

*3.6. Impact between Domains.* We can analyse the impact of a domain to other domains as Table 4 shows. In this table, we compared the CDCL (the accuracy of predicting correct domains and correct labels) of 5 domains with that of 4

Table 2: Evaluation based on CDCL (the accuracy of predicting correct domains and correct labels).

| Methods | CIFAR-10 (%) | CIFAR-100 (%) | Mini-ImageNet (%) | EuroSAT (%) | Intel image classification (%) |
|---|---|---|---|---|---|
| Maximum appeared | 76.28 | 37.75 | 48.00 | 54.07 | 51.50 |
| Fusion method [19] | 69.98 | 38.22 | 59.26 | 88.93 | 80.87 |
| CMS-CMM | 80.30 | 57.99 | 66.37 | 89.46 | 63.27 |
| CMS-CMM-opt | 89.21 | 70.74 | 71.39 | 95.43 | 86.86 |

Table 3: Evaluation of domain scalability based on CDCL.

| Methods | CIFAR-10 (%) | CIFAR-100 | Mini-ImageNet | EuroSAT | Intel image classification |
|---|---|---|---|---|---|
| 1 domain | 92.76 | — | — | — | — |
| 2 domains | 89.21 | 72.43% | — | — | — |
| 3 domains | 89.21 | 72.28% | 72.13% | — | — |
| 4 domains | 89.21 | 71.12% | 71.62% | 95.43% | — |
| 5 domains | 89.21 | 70.74% | 71.39% | 95.43% | 86.86% |

Table 4: Impact between domains based on CDCL.

| Methods | CIFAR-10 | CIFAR-100 | Mini-ImageNet | EuroSAT | Intel image classification |
|---|---|---|---|---|---|
| Without CIFAR-10 | — | +2.68% | +1.63% | +0.00% | +0.00% |
| Without CIFAR-100 | +2.79% | — | +8.13% | +0.48% | +1.32% |
| Without Mini-ImageNet | +0.00% | +0.15% | — | +0.03% | +4.32% |
| Without EuroSAT | +0.00% | +1.16% | +0.52% | — | +0.00% |
| Without intel Image classification | +0.00% | +0.38% | +0.23% | +0.00% | — |

domains, which means we drop one domain to evaluate the relation between this domain and other ones. When we drop CIFAR-10, we found the accuracy of CIFAR-100 is increased more than that of others. By the same way, we can find relations between domains. When there are similar labels between domains, the prediction of domain and labels may easily go wrong. For example, the label "fox" of CIFAR-100 is similar to the label "white fox" of Mini-ImageNet. Thus, how to consider the similarity between datasets are important to the increase of accuracy.

*3.7. Evaluation on the Number of Models.* In this subsection, we evaluate the relation between the number of models and CDCL (the accuracy of predicting correct domains and correct labels). We set the number of models is from 2 to 6. As there may be different combinations of models, we evaluate the average accuracy of these combinations. As we can see in Figure 5, the accuracy of all datasets increases as the number of models becomes bigger. On the other side, when the number of models is 6, the accuracy of some datasets becomes lower than that of 5 models. When there are low accurate models, these may lower the classification accuracy of our framework. The proper number of models can be computed by the validation set.

*3.8. Evaluation by More Metrics on All Testing Samples.* As we can see in Table 5, our methods achieved better performance in the most of metrics. In the CDWL (the percentage of predicting correct domains and wrong labels) case, the percentages of our methods are higher than those of the other methods. This is because our methods can predict more correct domains, which may cause more wrong labels.

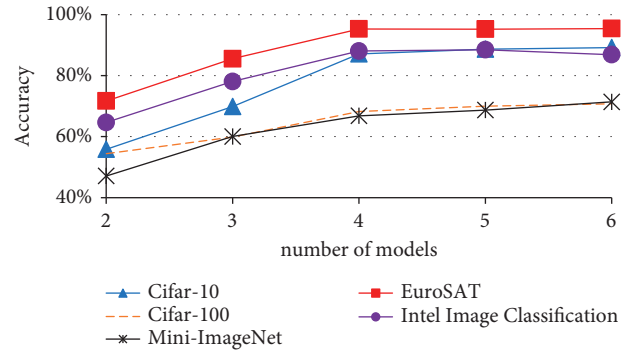


Figure 5: The evaluation on the number of models based on CDCL.

Compared with the fusion method [19], our method can increase 19.14% of CDCL while only increasing 9.01% CDWL.

*3.9. Evaluation of Execution Time and Memory Consumption.* Table 6 shows the total execution time and maximum memory consumption of each trained model on the corresponding dataset. We use Tesla K80 of NVIDIA [41] to run the models. In more detail, we use Tesla K80 of NVIDIA to run the model VoVNet-57 on CIFAR-10 and record the total execution time and maximum memory consumption of this model, which is shown in this table. Then, we also can use Tesla K80 of NVIDIA to run the model VoVNet-57 on the other datasets and record the total execution time and maximum memory consumption of this model. By the same pattern, we can run the other models on each dataset and record the total execution time and maximum memory consumption of these models.

Table 5: Evaluation by more metrics on all testing samples.

| Methods | CD: correct domains | CDCL: correct domains and correct labels | CDWL: correct domains and wrong labels | WD: wrong domains | WDCL: wrong domains and correct labels | WDWL: wrong domains and wrong labels |
|---|---|---|---|---|---|---|
| Maximum appeared | 55.98% | 53.49% | 2.49% | 44.02% | 32.36% | 11.66% |
| Fusion method [19] | 62.85% | 60.72% | 2.13% | 37.15% | 25.13% | 12.02% |
| CMS-CMM | 77.70% | 70.35% | 7.35% | 22.30% | 15.51% | 6.79% |
| CMS-CMM-opt | 91.00% | 79.86% | 11.14% | 9.00% | 5.97% | 3.03% |

Table 6: Execution time and memory consumption.

| Method | Dataset | | | | | |
|---|---|---|---|---|---|---|
| | The execution time (second) | | | | | Max memory consumption (MB) |
| | CIFAR-10 | CIFAR-100 | Mini-ImageNet | EuroSAT | Intel image classification | |
| VoVNet-57 | 106.05 | 104.95 | 475.35 | 60.20 | 46.63 | 136.25 |
| Res2Net50 | 105.20 | 396.89 | 474.80 | 59.37 | 44.04 | 91.01 |
| RepVGG-A0 | 34.67 | 35.55 | 45.69 | 20.70 | 15.93 | 30.35 |
| DenseNet121 | 85.24 | 376.06 | 451.03 | 48.91 | 36.65 | 27.52 |
| VGG16 | 122.20 | 121.59 | 495.97 | 72.79 | 55.64 | 513.73 |
| ResNet50 | 80.16 | 373.67 | 449.14 | 46.78 | 35.06 | 90.46 |
| CMS-CMM-opt in serial | 604.75 | 1479.95 | 2472.50 | 369.07 | 294.28 | 687.04 |
| CMS-CMM-opt in parallel | 222.52 | 497.24 | 600.29 | 164.02 | 146.87 | 839.29 |

Our methods generate multiple models on each dataset, which causes the runtime our methods become bigger than those of using single model. *CMS-CMM-opt in serial* runs the model one by one, which causes the execution time equals to the following: the execution time of single model × the number of models + the execution time of our fusion process. In more detail, our *CMS-CMM-opt in serial* run multiple models (one after one) on CIFAR-10. Then we run our fusion method. During these processes, we record the total execution time and maximum memory consumption that is shown in Table 6.

A simple solution to reduce the execution time is that we can use less models but this may lower the accuracy. To further reduce the execution time without lowing the accuracy, we run the models on the distributed computational nodes of a cluster based on the parallel pattern of paper [42]. In more detail, *CMS-CMM-opt in parallel* utilizes multiple computational nodes where each node has a Tesla K80 of NVIDIA [41]. As each node can run the model at the same time, the total execution time can be reduced. The total execution time is recorded as the end of all nodes and the maximum memory consumption is counted as the maximum one among these nodes. As *CMS-CMM-opt in parallel* of Table 6 indicates, the total runtime is reduced compared with *CMS-CMM-opt in serial*. The additional execution time is caused by the communication and our fusion process. The additional memory consumption is caused by the buffers of communication and our fusion process.

To achieve higher accuracy by single model on multiple domains, the structure becomes deeper and more complex which causes the memory consumption becomes bigger. For example, the V-MoE of Google achieves high accuracy with a trained model of 15 billion parameters on the ImageNet [43]. Compared with super model solution, our framework is more scalable.

*3.10. Illustration of the Domain and Label Classification.* Firstly, we use an example to explain the domain classification. As Figure 6 shows, we select the testing samples of label 1 that belongs to domain CIFAR-10. Then, we run all of the models on this sample and compute the average difference between the outputs of trained models that belong to the same datasets. Figure 6(a) is the difference of the outputs between the models that are trained on CIFAR-10. Figures 6(a)–6(e) are the differences of the outputs between the models that are trained on CIFAR-100, Mini-ImageNet, EuroSAT, and Intel image classification. As the testing samples belong to the CIFAR-10, the difference between the trained models of CIFAR-10 is obviously smaller than those of the other datasets. Thus, the domain classification based on model difference is reasonable.

We present the statistical result of model difference by the following. For a sample $S_n$ that belongs to the domain $D_u$, we use $\text{avgDM}(S_n \in D_u)$ to present the average difference between the trained models of $D_u$ as below:

$$\text{avgDM}(S_n \in D_u) = \frac{\left(\sum_{S_n \in D_u, u, j \neq 0} \| M_{0,u}(S_n) - M_{j,u}(S_n) \|_{\{L_k\}}\right)}{\left(\sum_{S_n \in D_u, u, j \neq 0, L_k} 1\right)}, \quad (13)$$

where $M_{0,u}(S_n) - M_{j,u}(S_n)_{\{L_k\}}$ is defined by (2), $M_{0,u}(S_n)$ is the output of highest accurate model, and $M_{j,u}(S_n)$ is the

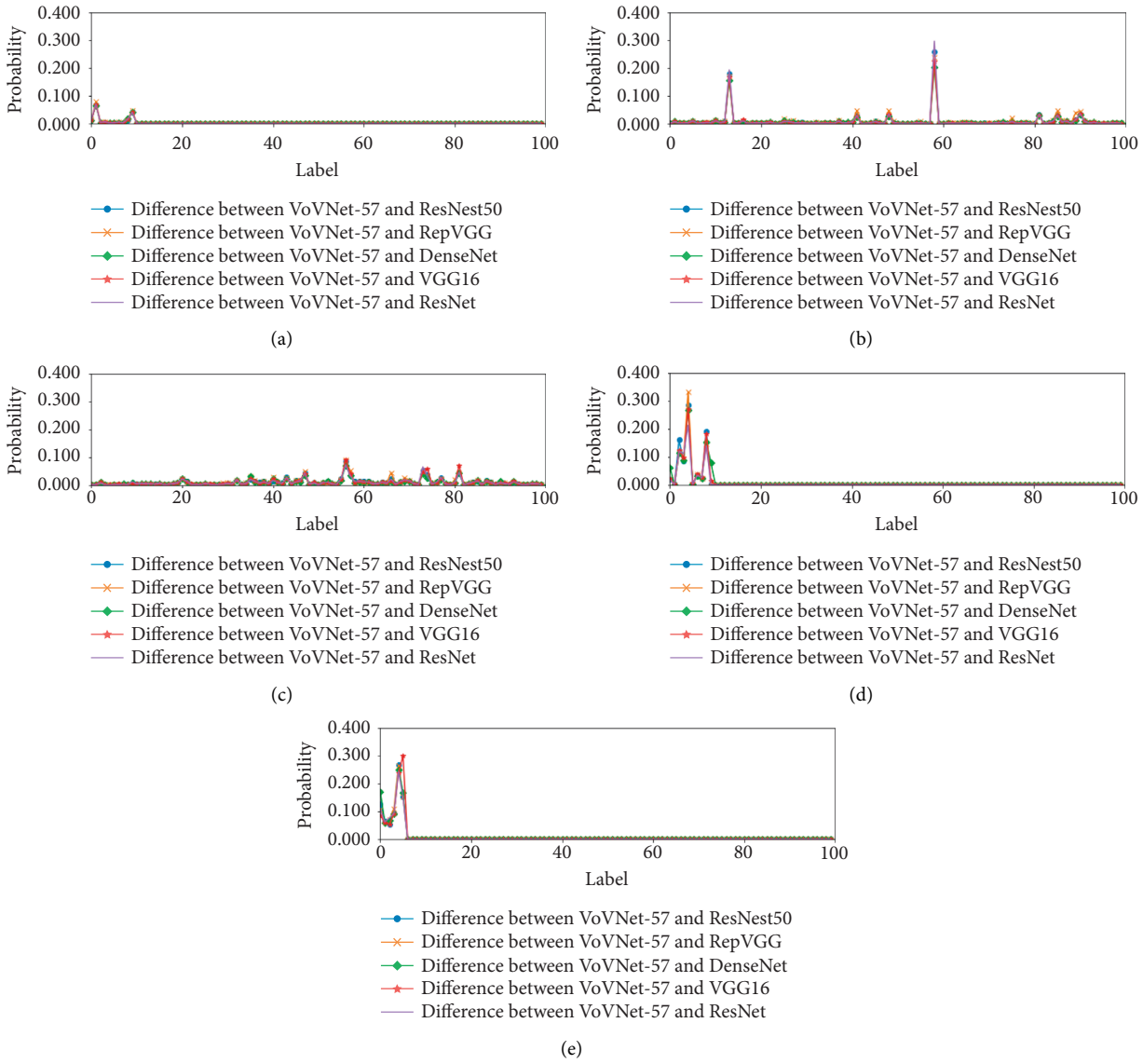


Figure 6: The difference between the output of trained models. The ground truth is label 1 sample of domain 0. (a) The models trained on CIFAR-10. (b) The models trained on CIFAR-100. (c) The models trained on Mini-ImageNet. (d) The models trained on EuroSAT. (e) The models trained on Intel image classification.

Table 7: Statistical analysis of model difference.

| Methods | CIFAR-10 | CIFAR-100 | Mini-ImageNet | EuroSAT | Intel image classification |
|---|---|---|---|---|---|
| avgDM$(S_n \in D_u)$ | 0.3093 | 0.7216 | 0.6468 | 0.3195 | 0.2487 |
| avgDM$(S_n \notin D_v)$ | 0.4860 | 1.211 | 1.5631 | 0.5841 | 0.3040 |

output of the other model. For a sample $S_n$ that belongs to the domain $D_u$, we use avgDM$(S_n \notin D_v)$ to present the difference between the models of $D_v$ $(v \neq u)$ as below:

$$\text{avgDM}(S_n \notin D_v) = \frac{\left(\sum_{S_n \in D_u,\, v, j \neq 0} \| M_{0,v}(S_n) - M_{j,v}(S_n) \|_{\{L_k\}}\right)}{\left(\sum_{S_n \in D_u,\, v, j \neq 0, L_k} 1\right)}, \tag{14}$$

where $M_{0,v}(S_n)$ is the output of highest accurate model on the validation samples of $D_v$, $M_{j,v}(S_n)$ is the output of the other model the validation samples of $D_v$. As we can see in Table 7, avgDM$(S_n \in D_u)$ is obviously smaller than avgDM$(S_n \notin D_v)$ on each dataset that means we can use this value the predict the domain. Based on this analysis, our methods further optimize the prediction of domain. CIFAR-100 and Mini-ImageNet has bigger number of labels than the other datasets, which causes the model difference bigger than that of the other datasets.

Secondly, we use three examples to explain the label classification based on the trained models of corresponding

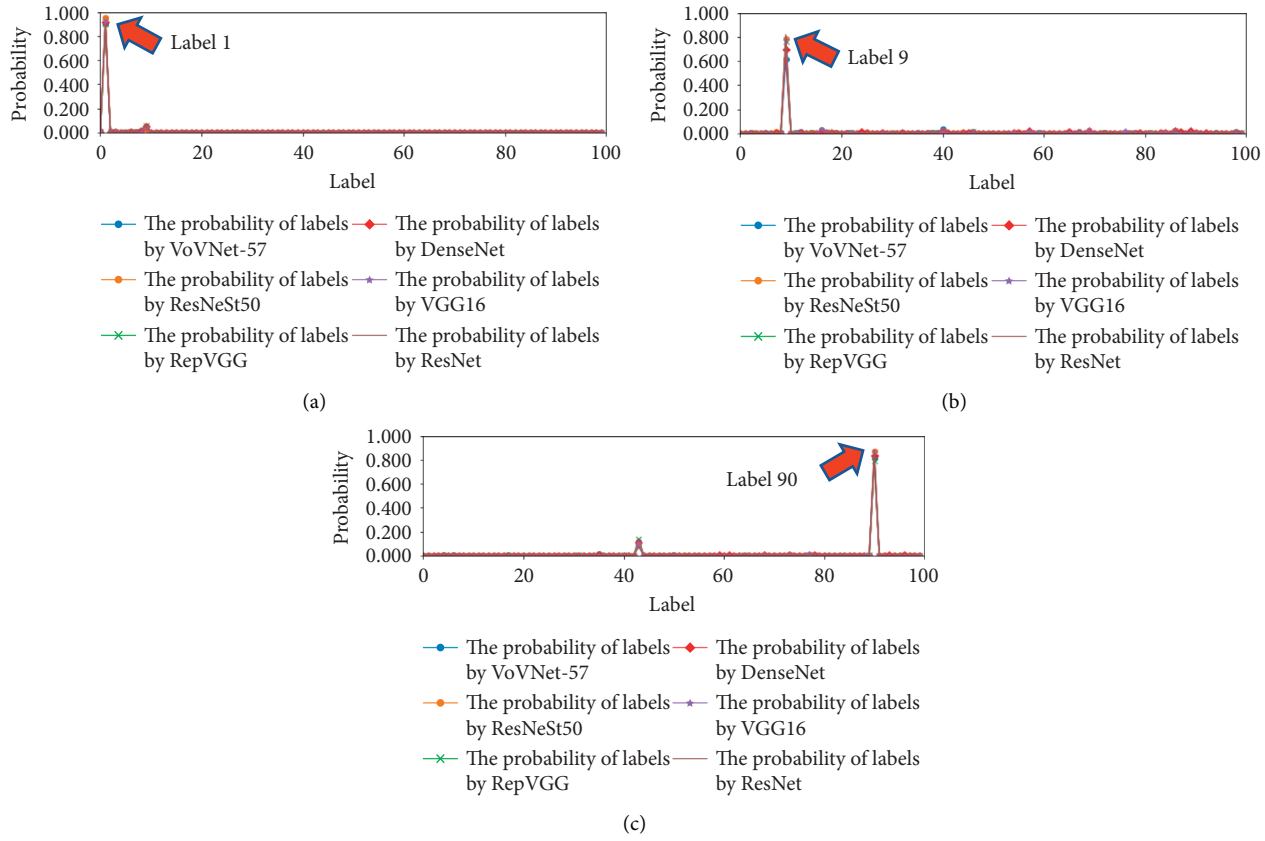


Figure 7: The probability of labels by different models of corresponding domain. (a) The ground truth is label 1 of domain 0. (b) The ground truth is label 9 of domain 1. (c) The ground truth is label 90 of domain 2.

Table 8: Statistical analysis of predicting label.

| Methods | CIFAR-10 | CIFAR-100 | Mini-ImageNet | EuroSAT | Intel image classification |
|---|---|---|---|---|---|
| $\text{avg}P(M_{i,u}(S_n) = L_{\text{groundtruth}})$ | 0.8101 | 0.5894 | 0.7074 | 0.8491 | 0.8588 |
| $\text{avg}P(M_{i,u}(S_n) \neq L_{\text{groundtruth}})$ | 0.0659 | 0.0749 | 0.0578 | 0.0552 | 0.0950 |

domain. In the first example, we select the testing samples of label 1 from CIFAR-10. Then, we run the trained models of CIFAR-10 on these samples. As Figure 7(a) shows, the average probability of label 1 by each model (trained on CIFAR-10) is obviously higher than those of the other labels. In Figure 7(b) case, we select the label 9 testing samples from CIFAR-100. Then, we run the trained models of CIFAR-100 on these samples. In Figure 7(c) case, we select label 90 testing samples from Mini-ImageNet. Then, we run the trained models of Mini-ImageNet on these samples. All of these cases show that the average probability of ground truth label is obviously higher than those of the other labels when we correctly select the trained models of corresponding dataset. Thus, the classification based on the probability of labels is reasonable.

We present the statistical analysis of label probability by the models. For a sample $S_n$ that belongs to the domain $D_u$, we define the average probability of ground truth label by the trained models of $D_u$ as below:

$$\text{avg}P\left(M_{i,u}(S_n) = L_{\text{groundtruth}}\right) = \frac{\left(\sum_{i,u,S_n} P\left(M_{i,u}(S_n) = L_{\text{groundtruth}}\right)\right)}{\sum_{i,u,S_n} 1}, \quad (15)$$

where $P(M_{i,u}(S_n) = L_{\text{groundtruth}})$ is introduced in equation (1). For a sample $S_n$ that belongs to the domain $D_u$, we define the average value of maximum probability of other label by the models as below:

Table 9: Introduction of the employed acronyms.

| Num | Acronyms | Introduction |
|---|---|---|
| 1 | CIFAR-10 | Dataset that is introduced in [32, 33]. |
| 2 | CIFAR-100 | Dataset that is introduced in [34, 35]. |
| 3 | Mini-ImageNet | Dataset that is introduced in [36, 37]. |
| 4 | EuroSAT | Dataset that is introduced in [38, 39]. |
| 5 | Intel image classification | Dataset that is introduced in [40]. |
| 6 | $S_n$ | Presents a sample. |
| 7 | $L_k$ | Presents a label. |
| 8 | $G_n$ | Presents the ground truth on $S_n$. |
| 9 | CMS-CMM | Our framework that is introduced in subsection 3.3 |
| 10 | CMS-CMM-opt | Our framework that is introduced in subsection 3.4 |
| 11 | Tesla K80 | NVIDIA GPU that is introduced in subsection 4.9 |
| 12 | CMS-CMM-opt in serial | Our serial execution that is introduced in subsection 4.9 |
| 13 | CMS-CMM-opt in parallel | Our parallel execution that is introduced in subsection 4.9 |
| 14 | VoVNet-57 | A deep learning model that is introduced in [20]. |
| 15 | ResNeSt50 | S deep learning model that is introduced in [21]. |
| 16 | RepVGG | A deep learning model that is introduced in [22]. |
| 17 | DenseNet | A deep learning model that is introduced in [23]. |
| 18 | VGG16 | A deep learning model that is introduced in [24]. |
| 19 | ResNet | A deep learning model that is introduced in [25]. |
| 20 | CD | The accuracy of predicting correct domains |
| 21 | CDCL | The accuracy of predicting correct domains and correct labels |
| 22 | CDWL | The percentage of predicting correct domains and wrong labels |
| 23 | WD | The percentage of predicting wrong domains |
| 24 | WDCL | The percentage of predicting wrong domains and correct labels |
| 25 | WDWL | The percentage of predicting wrong domains and wrong labels |

$$\text{avg}P\left(M_{i,u}\left(S_n\right) \neq L_{\text{groundtruth}}\right) = \frac{\left(\sum_{i,u,S_n} \text{Maximum}_{L_k \neq L_{\text{groundtruth}}} P\left(M_{i,u}\left(S_n\right) = L_k\right)\right)}{\sum_{i,u,S_n} 1}. \tag{16}$$

As we can see in Table 8, $\text{avg}P\left(M_{i,u}\left(S_n\right) = L_{\text{groundtruth}}\right)$ is obviously bigger than $\text{avg}P\left(M_{i,u}\left(S_n\right) \neq L_{\text{groundtruth}}\right)$ on each dataset that means we can use this value to predict the label. Based on this analysis, our methods further optimize the prediction of label.

*3.11. Introduction of the Employed Acronyms.* We use Table 9 to give the introduction of the employed acronyms in this article for reader's convenience.

## 4. Conclusions

In this article, we have introduced a novel framework that achieves the scalability of classification by using multiple models. Different from the existing single super model methods, our framework lowers the consumption of computational resource and achieves good scalability at the same time. Furthermore, we solve the problem of existing fusion methods. Our framework can be a good solution for the applications, which has to classify more domains of samples.

In the future work, we will do research about how to solve the problem of similarity between domains and labels. In some cases, the similarity is caused by the similar labels of different domains like the "fox" in CIFAR-10 and "white fox" in CIFAR-100. In other cases, this may be caused by the similar features between labels of the same domain, which is related to the accuracy of models. We believe that these factors are the key of increasing the accuracy of classification.

## Data Availability

The data used in this study are available at CIFAR-10: https://tensorflow.Google.cn/datasets/catalog/cifar10, CIFAR-100: https://tensorflow.Google.cn/datasets/catalog/cifar100, Mini-ImageNet: https://github.com/topics/miniimagenet, EuroSAT: https://tensorflow.Google.cn/datasets/catalog/eurosat, and Intel image classification: https://www.kaggle.com/datasets/puneet6060/intel-image-classification.

## Conflicts of Interest

The authors declare that they have no conflicts of interest.

## Acknowledgments

This work has been supported by the National Natural Science Foundation of China (Grant nos. 61802279, 6180021345, 61702281, and 61702366), Natural Science Foundation of Tianjin (Grant nos. 18JCQNJC70300, 19JCTPJC49200, 19PTZWHZ00020, and 19JCYBJC15800), the Fundamental Research Funds for the Tianjin

Universities (Grant no. 2019KJ019), and The Tianjin Science and Technology Program (Grant no. 19PTZWHZ00020) and in part by the State Key Laboratory of ASIC and System (Grant nos. 2021KF014 and 2021KF015) and Tianjin Educational Commission Scientific Research Program Project (Grant nos. 2020KJ112 and 2018KJ215) and the fund of Beijing Polytechnic (Grant no. 2022X017-KXZ).

## References

[1] T. Kau, M. Ziurlys, M. Taschwer, A. Kloss-Brandstätter, G. Grabner, and H. Deutschmann, "Fda-approved deep learning software application versus radiologists with different levels of expertise: detection of intracranial hemorrhage in a retrospective single-center study," *Neuroradiology*, vol. 64, no. 5, pp. 981–990, 2022.

[2] Q. Gong, P. Wang, and Z. Cheng, "A data-driven model framework based on deep learning for estimating the states of lithium-ion batteries," *Journal of the Electrochemical Society*, vol. 169, no. 3, Article ID 30532, 2022.

[3] L. M. Hondelink, M. Hüyük, P. E. Postmus et al., "Development and validation of a supervised deep learning algorithm for automated whole-slide programmed death-ligand 1 tumour proportion score assessment in non-small cell lung cancer," *Histopathology*, vol. 80, no. 4, pp. 635–647, 2022.

[4] Y. Deng, L. Wang, C. Zhao et al., "A deep learning-based approach to automatic proximal femur segmentation in quantitative ct images," *Medical, & Biological Engineering & Computing*, vol. 60, no. 5, pp. 1417–1429, 2022.

[5] G. Jin, Y. Hu, Y. Jiao, J. Wen, and Q. Song, "Improving the performance of deep learning model-based classification by the analysis of local probability," *Complexity*, vol. 2021, no. 1, 11 pages, Article ID 5534385, 2021.

[6] Z. Farhoudi and S. Setayeshi, "Fusion of deep learning features with mixture of brain emotional learning for audio-visual emotion recognition," *Speech Communication*, vol. 127, no. 1, pp. 92–103, 2021.

[7] J. Son, J. Cha, Y. J. Moon et al., "Generation of He i 1083 nm images from SDO AIA images by deep learning," *The Astrophysical Journal*, vol. 920, no. 2, pp. 101–111, 2021.

[8] H. Liu, H. Xiong, Y. Wang, H. An, D. Dou, and D. Wu, "Exploring the common principal subspace of deep features in neural networks," *Machine Learning*, vol. 111, no. 3, pp. 1125–1157, 2022.

[9] M. Ai, Y. Xie, Z. Tang, J. Zhang, and W. Gui, "Deep learning feature-based setpoint generation and optimal control for flotation processes," *Information Sciences*, vol. 578, no. 1, pp. 644–658, 2021.

[10] M. Nasir, A. R. Javed, M. A. Tariq, M. Asim, and T. Baker, "Feature engineering and deep learning-based intrusion detection framework for securing edge iot," *The Journal of Supercomputing*, vol. 78, no. 6, pp. 8852–8866, 2022.

[11] A. Albayrak, "Classification of analyzable metaphase images using transfer learning and fine tuning," *Medical, & Biological Engineering & Computing*, vol. 60, no. 1, pp. 239–248, 2021.

[12] N. Gupta and A. S. Jalal, "Traditional to transfer learning progression on scene text detection and recognition: a survey," *Artificial Intelligence Review*, vol. 55, no. 4, pp. 3457–3502, 2022.

[13] C. Long, E. Jo, and Y. Nam, "Development of a yoga posture coaching system using an interactive display based on transfer learning," *The Journal of Supercomputing*, vol. 78, no. 4, pp. 5269–5284, 2022.

[14] R. K. Agrawal and A. Juneja, "Deep Learning Models for Medical Image Analysis: Challenges and Future Directions," in *Proceedings of the Big Data Analytics, 7th International Conference, BDA 2019*, Ahmedabad, India, December 2019.

[15] C. Chen, P. Zhang, H. Zhang et al., "Deep learning on computational-resource-limited platforms: a survey," *Mobile Information Systems*, vol. 2020, Article ID 8454327, 19 pages, 2020.

[16] I. Karabayir, O. Akbilgic, and N. Tas, "A novel learning algorithm to optimize deep neural networks: evolved gradient direction optimizer (EVGO)," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 1, no. 1, pp. 1–10, 2020.

[17] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, "Multi-classification approaches for classifying mobile app traffic," *Journal of Network and Computer Applications*, vol. 103, no. 1, pp. 131–145, 2018.

[18] Y. You, L. Fan, and X. Li, "On redundant weighted voting systems with components having stochastic arrangement increasing lifetimes," *Operations Research Letters*, vol. 49, no. 5, pp. 777–784, 2021.

[19] X. Liu and A. G. Richardson, "Edge deep learning for neural implants: a case study of seizure detection and prediction," *Journal of Neural Engineering*, vol. 18, no. 4, 2021.

[20] Y. Lee, J. W. Hwang, S. Lee, Y. Bae, and J. Park, "An energy and GPU-computation efficient backbone network for real-time object detection, " 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)," *IEEE*, vol. 2019, no. 1, pp. 752–760, 2019.

[21] S. Wang, Y. Liu, Y. Qing, C. Wang, T. Lan, and R. Yao, "Detection of Insulator defects with improved ResNeSt and region proposal network," *IEEE Access*, vol. 8, no. 1, 2020.

[22] X. Feng, X. Gao, and L. Luo, "X-SDD: a new benchmark for hot rolled steel strip surface defects detection," *Symmetry*, vol. 13, no. 4, pp. 706–721, 2021.

[23] Z. Zhang, X. Liang, X. Dong, Y. Xie, and G. Cao, "A sparse-view CT reconstruction method based on combination of DenseNet and deconvolution," *IEEE Transactions on Medical Imaging*, vol. 37, no. 6, pp. 1407–1417, 2018.

[24] Z. Liu, J. Wu, L. Fu et al., "Improved kiwifruit detection using pre-trained VGG16 with RGB and NIR information fusion," *IEEE Access*, vol. 8, no. 1, pp. 2327–2336, 2020.

[25] S. Zhao, L. Qiu, P. Qi, and Y. Sun, "A novel image classification model jointing attention and ResNet for scratch," *IWCMC*, vol. 1, no. 1, pp. 1498–1503, 2020.

[26] Y. Chen, C. L. Yang, Y. Zhang, and Y. Z. Li, "Deep conditional adaptation networks and label correlation transfer for unsupervised domain adaptation," *Pattern Recognition*, vol. 98, no. 1, Article ID 107072, 2020.

[27] J. Chen, M. Yang, and J. Ling, "Attention-based label consistency for semi-supervised deep learning based image classification," *Neurocomputing*, vol. 453, no. 11, pp. 731–741, 2021.

[28] S. Kadry, V. Rajinikanth, D. Taniar, R. Damaševičius, and X. P. B. Valencia, "Automated segmentation of leukocyte from hematological images—a study using various cnn schemes," *The Journal of Supercomputing*, vol. 78, no. 5, pp. 6974–6994, 2022.

[29] C. Xu, D. Liang, Y. Xu et al., "Autoscale: learning to scale for crowd counting," *International Journal of Computer Vision*, vol. 130, no. 2, pp. 405–434, 2022.

[30] M. A. Elaskily, M. H. Alkinani, A. Sedik, and M. M. Dessouky, "Deep learning based algorithm (ConvLSTM) for copy move forgery detection," *Journal of Intelligent and Fuzzy Systems*, vol. 40, no. 3, pp. 4385–4405, 2021.

[31] M. Gupta, L. Bhargava, and S. Indu, "Deep neural network learning for power limited heterogeneous system with workload classification," *Computing*, vol. 104, no. 1, pp. 95–122, 2021.
[32] N. Miloevi and M. Rackovi, "Synergy between traditional classification and classification based on negative features in deep convolutional neural networks," *Neural Computing & Applications*, vol. 33, no. 8, pp. 1–10, 2021.
[33] Z. Wang, T. Luo, M. Li, J. T. Zhou, R. S. M. Goh, and L. Zhen, "Evolutionary multi-objective model compression for deep neural networks," *IEEE Computational Intelligence Magazine*, vol. 16, no. 3, pp. 10–21, 2021.
[34] G. Lagani, F. Falchi, C. Gennaro, and G. Amato, "Comparing the performance of Hebbian against backpropagation learning using convolutional neural networks," *Neural Computing & Applications*, vol. 34, no. 8, pp. 6503–6519, 2022.
[35] T. Choudhary, V. Mishra, A. Goswami, and J. Sarangapani, "Heuristic-based automatic pruning of deep neural networks," *Neural Computing & Applications*, vol. 34, no. 6, pp. 4889–4903, 2022.
[36] X. Liu, F. Zhou, J. Liu, and L. Jiang, "Meta-learning based prototype-relation network for few-shot classification," *Neurocomputing*, vol. 383, no. 1, pp. 224–234, 2020.
[37] H. J. Ye, H. Hu, and D. C. Zhan, "Learning adaptive classifiers synthesis for generalized few-shot learning," *International Journal of Computer Vision*, vol. 129, no. 6, pp. 1930–1953, 2021.
[38] M. A. Günen, "Performance comparison of deep learning and machine learning methods in determining wetland water areas using EuroSat dataset," *Environmental Science and Pollution Research*, vol. 29, no. 14, Article ID 21092, 2021.
[39] R. Bharti, D. Saini, and R. Malik, "A novel approach for hyper spectral images using transfer learning," *IOP Conference Series: Materials Science and Engineering*, vol. 1022, no. 1, 2021.
[40] Kaggle, "Datasets," https://www.kaggle.com/datasets/puneet6060/intel-image-classification.
[41] Z. Ul Abideen, M. Ghafoor, K. Munir et al., "Uncertainty assisted robust tuberculosis identification with Bayesian convolutional neural networks," *IEEE Access*, vol. 2020, no. 8, 2020.
[42] X. Jin and H. N. Kim, "Parallel deep learning detection network in the MIMO channel," *IEEE Communications Letters*, vol. 24, no. 1, pp. 126–130, 2020.
[43] Googleblog, "Scaling vision with sparse mixture of experts," https://ai.googleblog.com/2022/01/scaling-vision-with-sparse-mixture-of.html.