applied sciences

MDPI

Article

CF-CNN: Coarse-to-Fine Convolutional Neural Network

Jinho Park, Heegwang Kim and Joonki Paik *

Graduate School of Advanced Imaging Science, Multimedia and Film Chung-Ang University, Seoul 06974, Korea; jhpark@ipis.cau.ac.kr (J.P.); heegwang@ipis.cau.ac.kr (H.K.)
* Correspondence: paikj@cau.ac.kr

Abstract: In this paper, we present a coarse-to-fine convolutional neural network (CF-CNN) for learning multilabel classes. The basis of the proposed CF-CNN is a disjoint grouping method that first creates a class group with hierarchical association, and then assigns a new label to a class belonging to each group so that each class acquires multiple labels. CF-CNN consists of one main network and two subnetworks. Each subnetwork performs coarse prediction using the group labels created by the disjoint grouping method. The main network includes a refine convolution layer and performs fine prediction to fuse the feature maps acquired from the subnetwork. The generated class set in the upper level has the same classification boundary to that in the lower level. Since the classes belonging to the upper level label are classified with a higher priority, parameter optimization becomes easier. In experimental results, the proposed method is applied to various classification tasks to show a higher classification accuracy by up to 3% with a much smaller number of parameters without modification of the baseline model.

Keywords: deep learning; convolutional neural network; image classification

Citation: Jinho, P.; Heegwang, K.; Joonki, P. CF-CNN: Coarse-to-Fine Convolutional Neural Network. *Appl. Sci.* **2021**, *11*, 3722. https://doi.org/10.3390/app11083722

Academic Editor: DaeEun Kim

Received: 21 March 2021
Accepted: 18 April 2021
Published: 20 April 2021

1. Introduction

Since AlexNet won the 2012 ImageNet Large Scale Visual Recognition Challenge (ISLVRC) with a quantum jump in the sense of recognition performance [1], various types of deep convolutional neural network (CNN) models have been proposed for many applications including feature extraction, image enhancement, computer vision, medical imaging, and network security, to name a few [2–8]. Most deep neural networks adopt a CNN model because of its localization property using efficient computation and end-to-end learning ability that can detect various features from the input image. Recently, CNN-based deep learning research has tended to scale up the network to solve nonlinear problems [9–14]. In general, the CNN-based methods scale up the depth of layers [9–12] and the number of filters in each layer [14,15].

Simonyan et al. stacked several convolutional filters to increase the depth while decreasing the size of feature map by increasing the size of pooling or stride [9]. The deep CNNs exhibited superior performance, and have been widely adopted with the control of the feature map size. Residual network introduced by He et al. adopted the concept of shortcut connections between residual units for residual learning to solve the gradient vanishing problem [16,17] in the deeper network architectures [11,12]. Zagoruyko et al. experimented on the relationship between the depth and width of the residual network in various ways to improve the performance, and then proposed the wide residual network (WRN) with a wider channel and reduced depth [14]. They also used the dropout method to prevent overfitting caused by many parameters [18].

PyramidNet gradually increases the dimension of feature maps with zero-padded shortcut connections to preserve the deep network architecture [15]. However, it is difficult to optimize parameters if the number of parameters of the network indefinitely increases.

To solve these problems, hierarchical deep CNN algorithms have been proposed to classify many classes into several categories by grouping related classes and then classifying

them using the CNN for each category. This method can improve the overall network performance by classifying a relatively small number of classes corresponding to each category. Wu et al., proposed CF-DRNet to classify five classes of diabetic retinopathy. CF-DRNet consists of a coarse network that determines the presence of diabetic retinopathy and a fine network of four severity grades of diabetic retinopathy. Finally, the grade of diabetic retinopathy is determined using the aggregation module [8]. Yan et al. proposed a hierarchical deep CNN (HD-CNN) model to classify categories and a sub-deep-CNN is applied to each category [19]. This method consists of (i) shared layers to extract low-level features that are shared across all subnetworks, (ii) a coarse category CNN to divide similar classes into a category, and (iii) an independent subnetwork for fine classification in each category. The final classification is performed based on weighted averaging by combining the coarse predictions obtained from the lower layer and the fine predictions obtained from each subnetwork. However, in this method, the number of required subnetworks is proportional to the number of categories and each subnetwork requires pretraining. Zhu and Bain proposed a branch convolutional neural network (B-CNN). B-CNN adds a subnetwork that performs coarse prediction while the main network performs both fine and coarse predictions from each subnetwork, and finally, obtains the classification result using the weight loss function [20]. Verma et al. proposed a three-stage hierarchical Yoga data set, and successfully validated the hierarchical data set by using a network similar to B-CNN except with a single fully connected layer [21]. Kim et al. proposed a hierarchical network model by grouping the weights of the network with high correlation with the class [22]. As a result, the proposed network structure is suitable for a distributed machine learning environment with a significantly reduced number of parameters while maintaining a similar performance. Figure 1 illustrates the network structure for classification with a hierarchical structure.

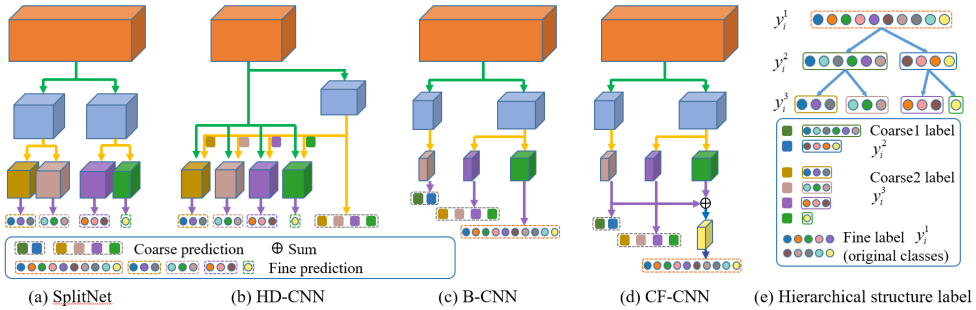

Figure 1. Network structure and hierarchical structure label illustrations. (**a**) SplitNet [22], (**b**) HD-CNN [19], (**c**) B-CNN [20], (**d**) our CF-CNN, and (**e**) hierarchical structure label.

In this paper, we present a hierarchical learning method, called coarse-to-fine convolutional neural network (CF-CNN). Figure 1d shows the concept of the proposed network. CF-CNN consists of a main network for fine classification and subnetworks for coarse prediction of classes. To predict a fine class, the last feature map of each subnetwork is used together with the last feature map of the main network through the refine convolution layers. Since a subnetwork of CF-CNN has a lower depth layer than the main network, the gradient vanishing problem can be alleviated in the learning process. The refine convolution layer fuses feature maps generated from subnetworks for coarse prediction. The feature map created from the subnetwork serves as a guide for the main network to perform finer classification. More accurate network parameter information is shown in Tables 1–7. In the experimental results section, we show that the proposed method can be applied to various CNN-based multilabel classification problems with an improved performance over existing methods. Our code will be made available at https://github.com/dkskzmffps/CF-CNN.

Table 1. Structure of the CF-VGG-16 model for CIFAR 100 dataset. Since the CIFAR image is very small, the last max-pooling layer of VGG-16 was not used.

CF-VGG-16						
Layer Name	**Output Size**	**Block Structure**	**Block Numbers**			
			Baseline	**Main**	**Coarse1**	**Coarse2**
Conv1	32 × 32	3 × 3, 64 3 × 3, 64	1	1		
Maxpool	16 × 16	2 × 2	1	1		
Conv2	16 × 16	3 × 3, 128 3 × 3, 128	1	1		
Maxpool	8 × 8	2 × 2	1	1		
Conv3	8 × 8	3 × 3, 256 3 × 3, 256 3 × 3, 256	1	1		
Maxpool	4 × 4	2 × 2	1	1		
Conv4	4 × 4	3 × 3, 512 3 × 3, 512 3 × 3, 512	1	1	1	
Maxpool	2 × 2	2 × 2	1	1	1	
Conv5	2 × 2	3 × 3, 512 3 × 3, 512 3 × 3, 512	1	1	1	1
Refine layer	2 × 2	3 × 3, 512 3 × 3, 512 3 × 3, 512		1		
fc1	4096		1	1		
fc2	4096		1	1	1	1
fc3	Classes number		1	1	1	1

Table 2. Structure of the CF-ResNet model for CIFAR-10 and CIFAR-100 datasets.

CF-ResNet						
Layer Name	**Output Size**	**Block Structure**	**Block Numbers 164-Layer (326-Layer)**			
			Baseline	**Main**	**Coarse1**	**Coarse2**
Conv1	32 × 32	3 × 3, 16	1 (1)	1 (1)		
Conv2	32 × 32	1 × 1, 16 3 × 3, 16 1 × 1, 64	18 (36)	18 (36)		
Conv3	16 × 16	1 × 1, 32 3 × 3, 32 1 × 1, 128	18 (36)	18 (36)	6 (6)	
Conv4	8 × 8	1 × 1, 64 3 × 3, 64 1 × 1, 256	18 (36)	18 (36)	6 (6)	6 (6)

Table 2. *Cont.*

CF-ResNet						
Layer Name	**Output Size**	**Block Structure**	**Block Numbers 164-Layer (326-Layer)**			
			Baseline	**Main**	**Coarse1**	**Coarse2**
Refine layer	8×8	1×1, 64 3×3, 64 1×1, 256		2 (2)		
Average pooling	1×1	8×8	1 (1)	1 (1)	1 (1)	1 (1)
fc	Classes number		1 (1)	1 (1)	1 (1)	1 (1)

Table 3. Structure of the CF-Pre-ResNet model for CIFAR-10 and CIFAR-100 datasets.

CF-Pre-ResNet						
Layer Name	**Output Size**	**Block Structure**	**Block Numbers 326-Layer (1001-Layer)**			
			Baseline	**Main**	**Coarse1**	**Coarse2**
Conv1	32×32	3×3, 16	1 (1)	1 (1)		
Conv2	32×32	1×1, 16 3×3, 16 1×1, 64	36 (111)	36 (111)		
Conv3	16×16	1×1, 32 3×3, 32 1×1, 128	36 (111)	36 (111)	6 (6)	
Conv4	8×8	1×1, 64 3×3, 64 1×1, 256	36 (111)	36 (111)	6 (6)	6 (6)
Refine layer	8×8	1×1, 64 3×3, 64 1×1, 256		2 (2)		
Average pooling	1×1	8×8	1 (1)	1 (1)	1 (1)	1 (1)
fc	Classes number		1 (1)	1 (1)	1 (1)	1 (1)

Table 4. Structure of the CF-Wide-ResNet-28layer model for CIFAR-10 and CIFAR-100 datasets.

CF-Wide-ResNet-28layer						
Layer Name	**Output Size**	**Block Structure**	**Block Numbers $k = 10$ ($k = 12$)**			
			Baseline	**Main**	**Coarse1**	**Coarse2**
Conv1	32×32	3×3, 16	1 (1)	1 (1)		
Conv2	32×32	3×3, $16 \times$ k 3×3, $16 \times$ k	4 (4)	4 (4)		
Conv3	16×16	3×3, $16 \times$ k 3×3, $16 \times$ k	4 (4)	4 (4)	1 (1)	
Conv4	8×8	3×3, $16 \times$ k 3×3, $16 \times$ k	4 (4)	4 (4)	1 (1)	2 (2)
Refine layer	8×8	3×3, $16 \times$ k 3×3, $16 \times$ k		1 (1)		
Average pooling	1×1	3×3, $16 \times$ k 3×3, $16 \times$ k	1 (1)	1 (1)	1 (1)	1 (1)

Table 4. *Cont.*

CF-Wide-ResNet-28layer						
Layer Name	**Output Size**	**Block Structure**	**Block Numbers $k = 10$ ($k = 12$)**			
			Baseline	**Main**	**Coarse1**	**Coarse2**
fc	Classes number		1 (1)	1 (1)	1 (1)	1 (1)

Table 5. Structure of the CF-PyramidNet model for CIFAR-10 and CIFAR-100 datasets.

CF-PyramidNet						
Layer Name	**Output Size**	**Block Structure**	**Block Numbers 110-Layer (200-Layer)**			
			Baseline	**Main**	**Coarse1**	**Coarse2**
Conv1	32×32	3×3, 16	1 (1)	1 (1)		
Conv2	32×32	$\begin{bmatrix} 1 \times 1, D_k \\ 3 \times 3, D_k \\ 1 \times 1, D_k \times 4 \end{bmatrix}$	$\alpha = 200$ $N = 36$ 12 (30)	$\alpha = 200$ $N = 36$ 12 (30)		
Conv3	16×16	$\begin{bmatrix} 1 \times 1, D_k \\ 3 \times 3, D_k \\ 1 \times 1, D_k \times 4 \end{bmatrix}$	$\alpha = 200$ $N = 36$ 12 (30)	$\alpha = 200$ $N = 36$ 12 (30)	$\alpha = 133$ $N = 12$ 6 (6)	
Conv4	8×8	$\begin{bmatrix} 1 \times 1, D_k \\ 3 \times 3, D_k \\ 1 \times 1, D_k \times 4 \end{bmatrix}$	$\alpha = 200$ $N = 36$ 12 (30)	$\alpha = 200$ $N = 36$ 12 (30)	$\alpha = 133$ $N = 12$ 6 (6)	$\alpha = 67$ $N = 6$ 6 (6)
Refine layer	8×8	$\begin{bmatrix} 1 \times 1, D_k \\ 3 \times 3, D_k \\ 1 \times 1, D_k \times 4 \end{bmatrix}$		2 (2)		
Average pooling	1×1	8×8	1 (1)	1 (1)	1 (1)	1 (1)
fc	Classes number		1 (1)	1 (1)	1 (1)	1 (1)

Table 6. Structure of the CF-ResNet and CF-Pre-ResNet model for ILSVRC 2012 dataset.

CF-ResNet and CF-Pre-ResNet						
Layer Name	**Output Size**	**Block Structure**	**Block Numbers 152-Layer**			
			Baseline	**Main**	**Coarse1**	**Coarse2**
Conv1	112×112	7×7, 64	1	1		
Average pooling	56×56	3×3	1	1		
Conv2	56×56	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix}$	3	3		
Conv3	28×28	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix}$	8	8	4	
Conv4	14×14	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix}$	36	36	6	6
Conv5	7×7	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix}$	3	3	3	3

Table 6. *Cont.*

CF-ResNet and CF-Pre-ResNet						
Layer Name	**Output Size**	**Block Structure**	**Block Numbers 152-Layer**			
			Baseline	**Main**	**Coarse1**	**Coarse2**
Refine layer	7×7	1×1, 512 3×3, 512 1×1, 2048	2			
Average pooling	1×1	7×7	1	1	1	1
fc	Classes number		1	1	1	1

Table 7. Structure of the CF-PyramidNet model for ILSVRC 2012 dataset.

CF-PyramidNet						
Layer Name	**Output Size**	**Block Structure**	**Block Numbers 200-Layer**			
			Baseline	**Main**	**Coarse1**	**Coarse2**
Conv1	112×112	3×3, 64	1	1		
Average pooling	56×56	3×3	1	1		
Conv2	56×56	1×1, D_k 3×3, D_k 1×1, $D_k \times 4$	$\alpha = 300$ $N = 66$ 3	$\alpha = 300$ $N = 66$ 3		
Conv3	28×28	1×1, D_k 3×3, D_k 1×1, $D_k \times 4$	$\alpha = 300$ $N = 66$ 24	$\alpha = 300$ $N = 66$ 24	$\alpha = 286$ $N = 13$ 4	
Conv4	14×14	1×1, D_k 3×3, D_k 1×1, $D_k \times 4$	$\alpha = 300$ $N = 66$ 36	$\alpha = 300$ $N = 66$ 36	$\alpha = 286$ $N = 13$ 6	$\alpha = 177$ $N = 9$ 6
Conv5	7×7	1×1, D_k 3×3, D_k 1×1, $D_k \times 4$	$\alpha = 300$ $N = 66$ 3	$\alpha = 300$ $N = 66$ 3	$\alpha = 286$ $N = 13$ 3	$\alpha = 177$ $N = 9$ 3
Refine layer	7×7	1×1, D_k 3×3, D_k 1×1, $D_k \times 4$		2		
Average pooling	1×1	7×7	1	1	1	1
fc	Classes number		1	1	1	1

Figure 2 shows the effect of the CF-CNN network structure on the CIFAR-100 dataset [23] compared with two standard preactivation ResNets with 326 layers and 1001 layers [12]. In CF-CNN, we have two subnetworks with 56 layers and refine the layers to preactivate ResNet with 326 layers and ResNet-1001, respectively, with an accuracy of 78.02% and 80.36%. The proposed CF-CNN has an accuracy of 80.77%. As a result, CF-CNN structure effectively improves the performance by adding a small number of layers instead of simply increasing the depth of the network.

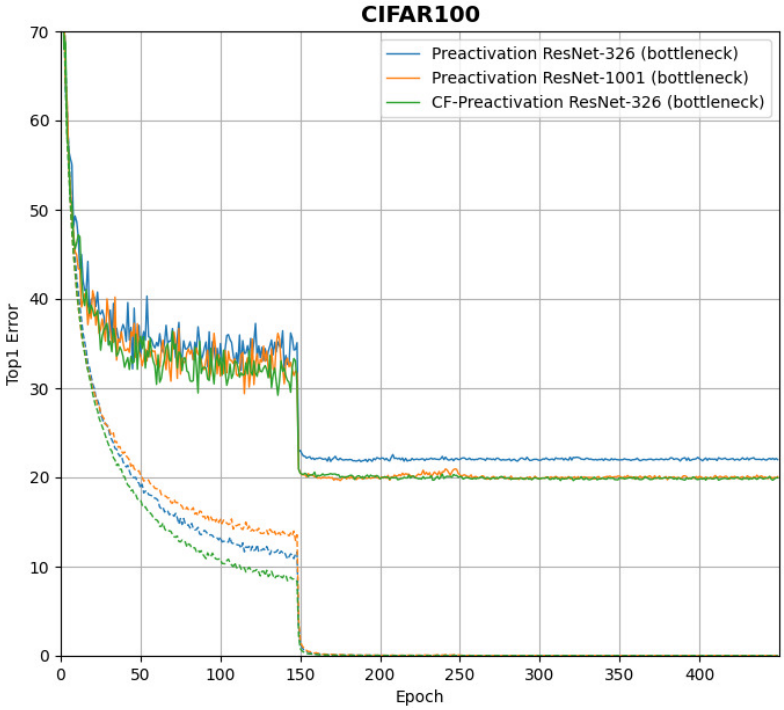

Figure 2. Comparison with the CIFAR-100 dataset.

2. Coarse-to-Fine Convolutional Neural Network

The hierarchically structured approach first divides multiple classes into several categories by grouping related classes, and then performs fine classification in each category using the CNN [19,22,24]. As a result, classification performance is improved at the cost of additional subnetworks and the corresponding learning method for each class group [19]. Although this approach may reduce the computational load, the classification accuracy is not preserved [22,24]. To solve that problem, the CF-CNN has the main network for fine classification and subnetworks for coarse classification. The proposed method uses the predicted class scores obtained from the baseline CNN model to group the classes, which belong to each label in the upper level with similar class scores to obtain new labels in the lower level. The created group label of each level is used as a classification label of each subnetwork, and both coarse and fine labels of each network are simultaneously trained.

2.1. Loss Function for CF-CNN

Figure 1d shows the architecture of the proposed CF-CNN. Given the group labels for the hierarchical structure, the labels in each hierarchical level are simultaneously trained by using the main and subnetworks. All feature maps of the last convolution layer in each subnetwork are used to predict fine classes via the refine layer. To obtain hierarchically structured group labels, we adopt disjoint grouping regularization proposed by Kim et al. [22]. Let $\boldsymbol{x}_i \in \mathbb{R}^d$ represent the input data instance, $\boldsymbol{y}_i \in \{1, ..., C\}$ the class label, and C the total number of classes. Given M training samples, $D = \{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{M}$, and corresponding class scores, $S = \{\boldsymbol{x}_i, \boldsymbol{s}_i\}_{i=1}^{M}$, obtained by the pretrained deep CNN, the goal of the disjoint grouping method is to obtain hierarchical multilevel labels $Q^l = \left\{\boldsymbol{x}_i, \boldsymbol{y}_i^l\right\}_{i=1}^{M}$ for coarse classification at subnetworks.

The class score vector, denoted as $\boldsymbol{s}_i \in \mathbb{R}^C$, is obtained by applying the softmax function to the deep CNN result, where C can be considered as the dimension of the class score. Let $\boldsymbol{y}_i^l \in \left\{1, ..., G^l\right\}$ and G^l represent the label and number of groups at the l-th hierarchy level, respectively. The label of the first level, that is $\boldsymbol{y}_i^1$, is equal to the original class label $\boldsymbol{y}_i$. To train the CF-CNN, we define the loss function as

$$\min_{W} \sum_{l=1}^{L} \mathcal{L}\left(\boldsymbol{W}, \boldsymbol{x}, \boldsymbol{y}^{l}\right), \tag{1}$$

where $\mathcal{L}\left(\boldsymbol{W}, \boldsymbol{x}, \boldsymbol{y}^{l}\right)$ represents the cross-entropy loss of hierarchically structured group labels at hierarchy level $l \in \{1, ..., L\}$ on the training data, L is the total number of hierarchy level, and $\boldsymbol{W}$ is the weight parameters of a network.

2.2. Disjoint Grouping Regularization

To obtain hierarchically structured group labels, we use the same disjoint grouping regularization, which was originally proposed by Kim et al. to divide classes belonging to the upper level group into lower level groups satisfying the disjoint property [22]. As shown in Figure 1a, Kim proposed a disjoint grouping regularization method to make the hierarchical network model by splitting the layers of the network. Kim's method creates a block diagonal weight matrix that belongs to a highly related class group by expressing the weights corresponding to each layer as a matrix. Since only the diagonal components assigned to the class group are learned during the learning process, the regularization process forces reducing the number of parameters to obtain a parallel model structure for distributed learning. On the other hand, we use this regularization with class scores from the pretrained model to generate hierarchical labels. Therefore, each class has a label having a hierarchical structure, and lower level classes have the same classification boundary with the upper level group.

Given the number of groups, denoted as K, let i represent the class belonging to the upper level label $\mathcal{G}$, then the binary variable p_{ki}^{g} indicates whether class i is assigned to a group k, $k = 1, ..., K$, or not. The disjoint group assignment vector of dimension K, denoted as $\boldsymbol{p}_{k}^{g}$, indicates whether the classes in the upper level label g are assigned to group k. Since our goal is to create a hierarchical label without duplication between classes, we assume that there is no overlap between groups, which results in $\sum_{k=1}^{K} \boldsymbol{p}_{k}^{g} = \mathbf{1}^{K}$, where $\mathbf{1}^{K}$ is the K vector of ones. Let $\boldsymbol{s}^{g}$ be the class scores belonging to label g, the proposed disjoint grouping method minimizes the combination of three objectives functions as

$$\min_{p} \mathcal{G}_{D}\left(\boldsymbol{s}^{g}, \boldsymbol{p}_{k}^{g}\right) + \lambda_{O}\mathcal{G}_{O}\left(\boldsymbol{p}_{k}^{g}\right) + \lambda_{B}\mathcal{G}_{B}\left(\boldsymbol{p}_{k}^{g}\right), \tag{2}$$

where λ_{O} and λ_{B} represent regularization parameters.

2.2.1. Disjoint Group Assignment

To apply the gradient descent optimization method, we change the binary variable p_{ki}^{g} to real variables in the range $[0, 1]$ with constraint $\sum_{k=1}^{K} \boldsymbol{p}_{k}^{g} = \mathbf{1}^{K}$. We use the softmax function to reparametrize p_{ki}^{g} with unconstrained variables z_{ki}.

$$p_{ki}^{g} = \frac{e^{z_{ki}}}{\sum_{k=1}^{K} e^{z_{ki}}}. \tag{3}$$

The objective function to create a class group satisfying the disjoint property is as follows.

$$\mathcal{G}_{D}\left(\boldsymbol{s}_{mean}^{g}, \boldsymbol{p}_{k}^{g}\right) = \sum_{k<j} \left(\sum_{i=1}^{C^{g}} \boldsymbol{s}_{i,mean}^{g} p_{ki}^{g} \cdot \sum_{i=1}^{C^{g}} \boldsymbol{s}_{i,mean}^{g} p_{ji}^{g} \right), \tag{4}$$

where $i \in \{1, ..., C^{g}\}$ represents a class belonging to upper level group g, C^{g} is the total number of classes in group g, and $\boldsymbol{s}_{i,mean}^{g}$ is the class score mean vector of the i class obtained from the pretrained baseline model. Kim's method aimed to learn the diagonal weight matrix using the feature assignment vector and the class assignment vector, whereas the proposed method aims to group the classes to satisfy the disjoint property using the class score.

2.2.2. Orthogonal Property

If we assume that there is no overlap between the groups, the group assignment vector should be orthogonal, i.e., $\boldsymbol{p}_k^g \cdot \boldsymbol{p}_j^g = 0$, $\forall i \neq j$. The group assignment vectors obtained by Equation (4) also exhibit orthogonal properties but add a regularization term to obtain better results.

$$\mathcal{G}_O\left(\boldsymbol{p}_k^g\right) = \sum_{k<j} \boldsymbol{p}_k^g \cdot \boldsymbol{p}_j^g. \tag{5}$$

2.2.3. Group Balance

The group assignment vector obtained by Equations (4) and (5) may assign most classes to one group. In an extreme case, all classes can be assigned to one group. To avoid that problem, we add a regularization term to control balance between groups. The corresponding regularization term is defined as

$$\mathcal{G}_B\left(\boldsymbol{p}_k^g\right) = \sum_{k=1}^{K} \left(\sum_{i=1}^{C^g} p_{ki} \right)^2. \tag{6}$$

Figure 3 shows the effect of the group balance regularization. Each color bar represents the group k, and the width of the bar represents the class ratio belonging to each group k. With large λ_B, the corresponding groups have similar ratio. On the other hand, if λ_B is small, the group ratio may be flexible. However, a very small λ_B makes almost all the classes belong to one group.

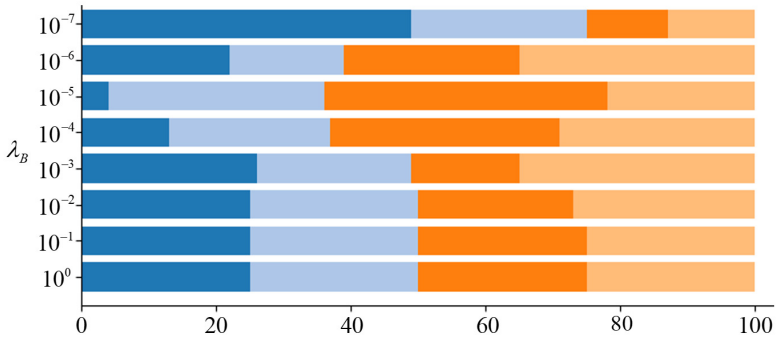

Figure 3. Effect of group balance regularization. The bar graph shows group assignment results for different values of λ_B with $K = 4$.

Figure 4 shows the result of creating a hierarchical label using the CIFAR-10 [23] dataset with 10 classes. To obtain the class score, we use the preactivation ResNet model [12] and parameters λ_O and λ_B are, respectively, set to 1 and 10^{-5}. In each subnetwork, the group labels at each level are used to predict coarse labels, and the feature map of the last convolution layer of each subnetwork is combined with the feature map of the convolution layer of the main network. The combined feature map is fused through a refine convolution layer and used for fine prediction.

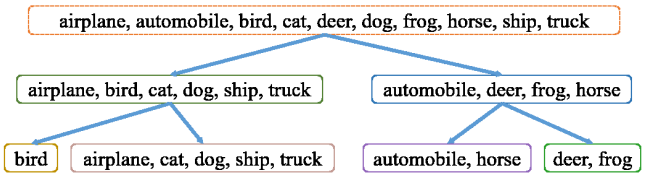

Figure 4. Result of the hierarchical structure label generation on the CIFAR-10 dataset.

3. Experimental Results

In this section, we evaluate the performance of the proposed CF-CNN. For the experiment, we tested the proposed method on various classification models including ResNet [11], WideResnet [14], preactivation ResNet [12], and PyramidNet [15] as the baseline models. The classification performance was evaluated on the CIFAR-10, CIFAR-100 [23], and ImageNet datasets [25]. Since the CIFAR-10, CIFAR-100, and ImageNet datasets contain the same number of data for each class, the method for the imbalance data problem [6,26] was not considered. In addition, in this experiment, data augmentation methods such as color transformation, geometric transformation, rotation, and contrast transformation were not used in order to focus on checking the performance difference between the proposed model and the baseline model [27–29].

Both CIFAR-10 and CIFAR-100 have 50,000 training and 10,000 test images. CIFAR-10 contains 10 classes and CIFAR-100 has 100 classes. In the training process, basic data augmentation such as horizontal flipping and padding as much as 4 pixels around the image and random cropping of 32×32 image were applied. Each model was trained using stochastic gradient descent (SGD), to which Nesterov momentum was applied for 400 epochs. The learning rate starts from 0.1 and decays by a factor of 10 at 150, 250, and 300 epochs. The batch size was 128. When training PyramidNet and CF-PyramidNet, the initial rate was 0.25, which decayed by a factor of 10 for every 120 epochs. The batch size was 64.

The ImageNet dataset includes 1000 classes and consists of one million training images and 50,000 validation images. For the experiment, we used 200 epochs to train each model, starting with a learning rate of 0.05 and decaying by a factor of 10 at 60, 90, and 120 epochs. The batch size was 64. The size of the image used for training and testing is 224 × 224. The learning process of the proposed CF-CNN consists of four steps: (i) training the baseline model using 90% of the training dataset; (ii) computing the class score of the remaining 10% of the training dataset using the trained network; (iii) generating multilabels using the disjoint grouping method; (iv) training the CF-CNN on the training dataset.

Table 8 shows results of classification using the baseline model and hierarchical structure labels obtained by various grouping methods. Manually divided hierarchical labels were generated using the method proposed by B-CNN and the number of classes for coarse 1 and coarse 2 were 8 and 20, respectively. In the case of the Random method, classes in the upper group were randomly selected and divided into 5 groups to form a hierarchical structure. The groups in each level had the same number of classes. For experimentation of the clustering method and the proposed method, we used the class score obtained from the pretrained baseline model. The clustering method used the k-means clustering method, and classes belonging to the upper group were divided into 5 groups using the k-means clustering method. For the disjoint grouping method, we set parameters λ_O and λ_B to 1 and 10^{-5}, respectively.

Table 8. Best classification accuracy according to the grouping method on CIFAR-100. (M), (R), and (C) denote *manual*, *random*, *clustering*, respectively.

Network Model	Number of Labels	Accuracy
Resnet-326 (baseline)	100	75.05
CF-Resnet-326 (M)	8, 20, 100	76.04
CF-Resnet-326 (R)	5, 25, 100	76.14
CF-Resnet-326 (C)	5, 25, 100	76.38
CF-Resnet-326 (Proposed)	5, 25, 100	76.85

The second column, called 'Number of labels', indicates the number of labels at each level including the number of labels used for coarse 1, coarse 2, and fine classification, respectively. The label used for fine classification represents the original label. The classification accuracy was evaluated using the CIFAR-100 dataset. When a hierarchical structure is created using the k-means method and the disjoint group method, classes with similar

characteristics form groups. When learning a subnetwork using such a group label, it shows better performance than the random grouping method or the manual grouping method because it is easier to find the optimal parameter in the process of fine prediction in the main network. The proposed method shows better performance than the k-means clustering method because it has stronger intergroup disjoint properties than the k-means method.

Tables 9–11 respectively summarize the classification accuracy for CIFAR-100, CIFAR-10, and ILSVRC 2012 datasets by applying the proposed method to various deep-learning models. B-CNN used the VGG-16 model as the baseline model [9]. HD-CNN adopted the NIN model [30], which doubled the number of filters in all convolutional layers in Table 9, and used VGG-16 model in Table 11. In Tables 9 and 11, SplitNet used WideResnet-16 (k = 8) and ResNet-18x2 models [22]. The rest of the models except for WideResNet used a bottleneck structure, detailed parameter information is shown in Tables 1–7. The parameter k of the WideResnet model represents a widening factor. In Pyramidnet, α and N represent the widening factor and the total number of blocks, respectively. In the proposed method, each subnetwork and main network have different parameter values for feature map fusion. D_k of Pyramidnet represents the channel dimensions of the k-th block. D_k is defined as

$$D_k = \begin{cases} 16 & \text{if } k = 1 \\ [D_{k-1} + \alpha/N] & \text{if } 2 \le k \le N+1. \end{cases} \quad (7)$$

Table 9. Classification results using various classification models and grouping labels on the CIFAR-100 dataset

CIFAR-100		
Network Model	**Number of Labels**	**Accuracy**
HD-CNN	9, 100	65.64
B-CNN	8, 20, 100	64.42
VGG-16	100	63.04
WideResnet-16 (k = 8)	100	75.74
SplitNet	100	76.04
ResNet-164	100	74.84
ResNet-326	100	75.05
Pre-ResNet-326	100	78.02
Pre-ResNet-1001	100	80.36
. WideResnet-28 (k = 10)	100	80.75
WideResnet-28 (k = 12)	100	81.48
PyramidNet-110 (α = 200)	100	81.98
PyramidNet-272 (α = 200)	100	84.36
CF-VGG-16	5, 25, 100	65.11
CF-ResNet-164	5, 25, 100	77.01
CF-ResNet-326	5, 25, 100	76.85
CF-Pre-ResNet-326	5, 25, 100	80.77
CF-Pre-ResNet-1001	5, 25, 100	82.09
CF-WideResnet-28 (k = 10)	5, 25, 100	82.38
CF-WideResnet-28 (k = 12)	5, 25, 100	82.67
CF-PyramidNet-110 (α = 200)	5, 25, 100	82.57
CF-PyramidNet-272 (α = 200)	5, 25, 100	84.94

In the proposed CF-CNN structure, the subnetwork for classifying each group label was created using the same layer structure used in each deep-learning model. Number of labels represents the number of group labels obtained using the disjoint grouping method, and represents the number of classes classified in coarse1, coarse2, and fine image classification, respectively. When compared with ResNet-326 in Table 9, CF-ResNet-164 performs better with a significantly smaller number of parameters. Likewise, compared

with Pre-ResNet-1001, CF-Pre-ResNet-326 performs better with a much smaller number of parameters for CIFAR-10 and ILSVRC 2012 datasets.

Table 10. Classification results with various classification models and grouping labels on the CIFAR-10 dataset.

CIFAR-10		
Network Model	**Number of Labels**	**Accuracy**
HD-CNN	–	–
B-CNN	2, 7, 10	88.22
SplitNet	–	–
Pre-ResNet-326	10	95.87
WideResnet-28 (k = 10)	10	95.83
WideResnet-28 (k = 12)	10	95.67
PyramidNet-272 (α = 200)	10	96.7
CF-Pre-ResNet-326	2, 4, 10	96.09
CF-WideResnet-28 (k = 10)	2, 4, 10	96.48
CF-WideResnet-28 (k = 12)	2, 4, 10	96.49
CF-PyramidNet-272 (α = 200)	2, 4, 10	96.3

Table 11. Classification results with various classification models and grouping labels on the ILSVRC 2012 dataset.

ILSVRC 2012		
Network Model	**Number of Labels**	**Accuracy**
HD-CNN	84, 1000	76.31
B-CNN	-	-
SplitNet	1000	75.1
ResNet-18x2	1000	74.42
ResNet-152	1000	77.0
Pre-ResNet-152	1000	77.8
PyramidNet-200 (α = 300)	1000	79.5
CF-ResNet-152	100, 487, 1000	78.4
CF-Pre-ResNet-152	100, 487, 1000	78.7
CF-PyramidNet-200 (α = 300)	100, 487, 1000	80.4

4. Conclusions

In this paper, we proposed a multilevel label augmentation method using a disjoint grouping method. We also proposed coarse-to-fine convolutional neural network (CF-CNN) to learn the generated multilevel label with a smaller set of network parameters. Multilevel labels created by the disjoint grouping method have a hierarchical structure and have the same classification boundary between levels. The CF-CNN has a subnetwork to simultaneously learn multilevel labels. In the experimental results, the proposed method was applied to various classification models. As a result, the proposed method shows better performance than existing models with a much smaller number of parameters, without requiring structural changes of the building blocks constituting the network.

Author Contributions: Conceptualization, (J.P.) Jinho Park and (J.P.) Joonki Paik; methodology, software, (J.P.) Jinho Park; validation, formal analysis, (J.P.) Jinho Park and H.K.; writing—Original draft preparation, (J.P.) Jinho Park; writing—Review and editing, H.K. and (J.P.) Joonki Paik; supervision, project administration, (J.P.) Joonki Paik. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by National R&D Program through the National Research Foundation of Korea(NRF) funded by Ministry of Science and ICT(2020M3F6A1110350) and the Institute of Civil-Military Technology Cooperation Program funded by the Defense Acquisition Program Administration and Ministry of Trade, Industry and Energy of Korean government under grant No. 19CM5119.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The CIFAR-10 and CIFAR-100 datasets presented in this study are openly available in [23] and can be found here https://www.cs.toronto.edu/~kriz/cifar.html. The ILSVRC2012 dataset can be found here http://image-net.org/challenges/LSVRC/2012/index.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish.

References

1. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. In Proceedings of the Advances in Neural Information Processing Systems (NIPS), Lake Tahoe, NV, USA, 3–6 December 2012; pp. 1106–1114.
2. Donahue, J.; Jia, Y.; Vinyals, O.; Hoffman, J.; Zhang, N.; Tzeng, E.; Darrell, T. Decaf: A deep convolutional activation feature for generic visual recognition. In Proceedings of the 31st International Conference on Machine Learning (ICML), Beijing, China, 21–26 June 2014; pp. 647–655.
3. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
4. Long, J.; Shelhamer, E.; Darrell, T. Fully Convolutional Networks for Semantic Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 8–10 June 2015; pp. 3431–3440.
5. Sermanet, P.; Eigen, D.; Zhang, X.; Mathieu, M.; Fergus, R.; LeCun, Y. OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks. In Proceedings of the 2nd International Conference on Learning Representations (ICLR), Banff, AB, Canada, 14–16 April 2014.
6. Chen, X.; Yang, Y.; Wang, S.; Wu, H.; Tang, J.; Zhao, J.; Wang, Z. Ship type recognition via a coarse-to-fine cascaded convolution neural network. *J. Navig.* **2020**, *73*, 813–832. [CrossRef]
7. Khan, M.A.; Kim, J. Toward Developing Efficient Conv-AE-Based Intrusion Detection System Using Heterogeneous Dataset. *Electronics* **2020**, *9*, 1771. [CrossRef]
8. Wu, Z.; Shi, G.; Chen, Y.; Shi, F.; Xinjian, C.; Coatrieux, G.; Yang, J.; Luo, L.; Li, S. Coarse-to-Fine Classification for Diabetic Retinopathy Grading using Convolutional Neural Network. *Artif. Intell. Med.* **2020**,*108*, 101936. [CrossRef] [PubMed]
9. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. In Proceedings of the 3rd International Conference on Learning Representations (ICLR), San Diego, CA, USA, 7–9 May 2015.
10. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.E.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going Deeper with Convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 1–9.
11. 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 (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
12. He, K.; Zhang, X.; Ren, S.; Sun, J. Identity Mappings in Deep Residual Networks. In Proceedings of the 14th European Conference on Computer Vision (ECCV), Amsterdam, The Netherlands, 11–14 October 2016; pp. 630–645.
13. Szegedy, C.; Ioffe, S.; Vanhoucke, V.; Alemi, A.A. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. In Proceedings of the AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 4–9 February 2017; pp. 4728–4284.
14. Zagoruyko, S.; Komodakis, N. Wide Residual Networks. In Proceedings of the British Machine Vision Conference (BMVC), York, UK, 19–22 September 2016.
15. Han, D.; Kim, J.; Kim, J. Deep Pyramidal Residual Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 6307–6315.

16. Bengio, Y.; Simard, P.; Frasconi, P. Learning Long-Term Dependencies with Gradient Descent is Difficult. *IEEE Trans. Neural Netw.* **1994**, *5*, 157–166. [CrossRef] [PubMed]
17. Glorot, X.; Bengio, Y. Understanding The Difficulty of Training Deep Feedforward Neural Networks. In Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS), Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010; pp. 249–256.
18. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
19. Yan, Z.; Zhang, H.; Piramuthu, R.; Jagadeesh, V.; DeCoste, D.; Di, W.; Yu, Y. HD-CNN: Hierarchical Deep Convolutional Neural Networks for Large Scale Visual Recognition. In Proceedings of the IEEE International Conference on Computer Vision 2015, Santiago, Chile, 7–13 December 2015; pp. 2740–2748.
20. Zhu, X.; Bain, M. B-CNN: Branch Convolutional Neural Network for Hierarchical Classification. *arXiv* **2017**, arXiv:1709.09890
21. Verma, M.; Kumawat, S.; Nakashima, Y.; Raman, S. Yoga-82: A New Dataset for Fine-Grained Classification of Human Poses. In Proceedings of the IEEE/CVF CVPR Workshops (CVPRW), Seattle, WA, USA, 14–19 June 2020; pp. 1038–1039.
22. Kim, J.; Park, Y.; Kim, G.; Hwang, S.J. SplitNet: Learning to Semantically Split Deep Networks for Parameter Reduction and Model Parallelization. In Proceedings of the 34th International Conference on Machine Learning (ICML), Sydney, NSW, Australia, 6–11 August 2017; pp. 1866–1874.
23. Krizhevsky, A. *Learning Multiple Layers of Features from Tiny Images*; Technical Report, University of Toronto: Toronto, ON, Canada, 2009. Available online: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.222.9220 (accessed on 20 April 2021).
24. Roy, D.; Panda, P.; Roy, K. Tree-CNN: A Hierarchical Deep Convolutional Neural Network for Incremental Learning. *Neural Netw.* **2020**, *121*, 148–160. [CrossRef] [PubMed]
25. Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. ImageNet Large Scale Visual Recognition Challenge. *Int. J. Comput. Vis.* **2015**, *115*, 211–252. [CrossRef]
26. Wang, Y.; Gan, W.; Yang, J.; Wu, W.; Yan, J. Dynamic Curriculum Learning for Imbalanced Data Classification. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea, 27 October–2 November 2019; pp. 5016–5025.
27. DeVries, T.; Taylor, G.W. Improved Regularization of Convolutional Neural Networks with Cutout. *arXiv* **2017**, arXiv:1708.04552.
28. Cubuk, E.D.; Zoph, B.; Mane, D.; Vasudevan, V.; Le, Q.V. AutoAugment: Learning Augmentation Strategies from Data. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 16–20 June 2019; pp. 113–123.
29. Cubuk, E.D.; Zoph , B.; Shlens, J.; Le, Q.V. Randaugment: Practical Automated Data Augmentation with a reduced Search Space. In Proceedings of the IEEE/CVF CVPR Workshops (CVPRW), Seattle, WA, USA, 14–19 June 2020; pp. 3008–3017.
30. Lin, M.; Chen, Q.; Yan, S. Network in network. *arXiv* **2013**, arXiv:1312.4400.