Decision Fusion Networks for Image Classification

Keke Tang, *Member, IEEE*, Yuexin Ma, Dingruibo Miao, Peng Song, Zhaoquan Gu, Zhihong Tian, *Senior Member, IEEE*, and Wenping Wang, *Fellow, IEEE*

***Abstract*— Convolutional neural networks, in which each layer receives features from the previous layer(s) and then aggregates/abstracts higher level features from them, are widely adopted for image classification. To avoid information loss during feature aggregation/abstraction and fully utilize lower layer features, we propose a novel decision fusion module (DFM) for making an intermediate decision based on the features in the current layer and then fuse its results with the original features before passing them to the next layers. This decision is devised to determine an auxiliary category corresponding to the category at a higher hierarchical level, which can, thus, serve as category-coherent guidance for later layers. Therefore, by stacking a collection of DFMs into a classification network, the generated decision fusion network is explicitly formulated to progressively aggregate/abstract more discriminative features guided by these decisions and then refine the decisions based on the newly generated features in a layer-by-layer manner. Comprehensive results on four benchmarks validate that the proposed DFM can bring significant improvements for various common classification networks at a minimal additional computational cost and are superior to the state-of-the-art decision fusion-based methods. In addition, we demonstrate the generalization ability of the DFM to object detection and semantic segmentation.**

***Index Terms*— Image classification, image recognition, network architecture, neural networks.**

Manuscript received 4 June 2020; revised 1 June 2021 and 15 February 2022; accepted 22 July 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 62102105, Grant U20B2046, and Grant 61902082; in part by the Guangdong Basic and Applied Basic Research Foundation under Grant 2020A1515110997 and Grant 2022A1515011501; in part by the Science and Technology Program of Guangzhou under Grant 202002030263 and Grant 202102010419; in part by the Open Project Program of the State Key Laboratory of CAD&CG, Zhejiang University, under Grant A2218; in part by the Guangdong Higher Education Innovation Group under Grant 2020KCXTD007; in part by the Guangzhou Higher Education Innovation Group under Grant 202032854; and in part by the Shanghai Sailing Program under Grant 22YF1428700. *(Corresponding author: Zhihong Tian.)*

Keke Tang and Zhihong Tian are with the Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou 510006, China (e-mail: tangbohutbh@gmail.com; tianzhihong@gzhu.edu.cn).

Yuexin Ma is with the School of Information Science and Technology, ShanghaiTech University, Shanghai 200031, China (e-mail: yuexinma93@gmail.com).

Dingruibo Miao is with the State Key Laboratory of Remote Sensing Information Engineering for Surveying and Mapping, Wuhan University, Wuhan 430072, China (e-mail: miaodrb@gmail.com).

Peng Song is with the Information Systems Technology and Design Pillar, Singapore University of Technology and Design, Singapore 487372 (e-mail: songpenghit@gmail.com).

Zhaoquan Gu is with the Department of Computer Science and Technology, Harbin Institute of Technology, Shenzhen 518055, China, and also with the Department of New Networks, Peng Cheng Laboratory, Shenzhen 518066, China (e-mail: demin456@gmail.com).

Wenping Wang is with the Department of Visualization, Texas A&M University, College Station, TX 77843 USA (e-mail: wenping@cs.hku.hk).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2022.3196129.

Digital Object Identifier 10.1109/TNNLS.2022.3196129

I. Introduction

IMAGE classification [1]–[6], with the aim of classifying an image into one of several predefined categories, is an essential problem in computer vision [7]. In the last few decades, researchers focus on representing images with handcrafted low-level descriptors [8]–[11] and then discriminating images based on these representations with a classifier (e.g., an support vector machine (SVM) [12] or one of its variants [13], [14]). However, due to a lack of high-level features, the performance tends to become saturated. Thanks to the availability of enormous labeled datasets [15] and powerful computational infrastructures, convolutional neural networks (CNNs) can automatically extract discriminative high-level features from training images, significantly improving the state-of-the-art performance [3], [16]–[18].

Although high-level features are more discriminative, utilizing them alone to classify images is still challenging since with a growing number of categories, the possibility of confusion increases. In addition, the low-level features extracted in the early layers can be used to separate groups of classes at higher hierarchical levels [19]. To avoid discarding such information during feature aggregation/abstraction, some researchers attempt to combine high- and low-level features to exploit their complementary strengths [20]. However, a simple combination method will cause the features to have relatively high dimensionality, hindering their practical use.

Other researchers employ low-level features to make coarse decisions and then utilize high-level features to make finer decisions by designing deep decision trees implemented with CNNs based on the divide-and-conquer approach. With a hierarchical structure of categories, a straightforward way is to use the networks of the root node to identify the coarsest category and then perform dynamic routing to the networks of a child node to recursively determine the finer categories [21]. However, hierarchical category information is not always available and may not reflect the difficulty of classification. Therefore, researchers are required to design suitable division solutions, making the training process extremely complex (e.g., with multiple stages). In addition, current deep decision tree-based methods face two fatal weaknesses: 1) the network should save all of the tree branches, causing the number of parameters to grow explosively larger than the number for a single classification network and 2) once a decision is routed to an incorrect path, it can be difficult to recover.

To resolve the above-mentioned issues, we propose a novel decision fusion module (DFM). The rationale is that if we

adopt the current layer to generate a category-coherent auxiliary decision to determine the category at a higher hierarchical level and then propagate it together with the original features to subsequent layers, then those layers can be guided to aggregate and abstract more discriminative features. By stacking collections of DFMs into backbone classification networks, the generated decision fusion networks (DFNets) are explicitly formulated to progressively encode more discriminative features guided by the auxiliary decisions made by previous layers and then iteratively refine those decisions based on the newly generated features. Inspired by residual learning [3], it is much easier to optimize the refinement process than to optimize the making of an unreferenced new decision from scratch. In addition, the properties of DFM enable DFNets to naturally overcome the weaknesses of common deep decision trees. First, in contrast to dynamic routing between several different branches after making a decision, DFM applies the decision as a conditional code for the next layers similar to [22], to enable routing without the creation of additional network branches. In addition, the conditional coding scheme enables DFMs in later layers to recover from some false decisions made previously. Furthermore, unlike traditional deep decision trees that designate each intermediate decision following a prespecified category hierarchy, DFMs can adaptively learn more appropriate category divisions under weak supervision provided by three novel loss functions, and they can be trained in a completely end-to-end fashion along with the backbone networks. Overall, our contribution is threefold.

1) We design a novel DFM that can generate and combine auxiliary decisions based on the current features as additional channels to guide subsequent layers.
2) We propose three novel loss functions to force DFM to make category-coherent auxiliary decisions.
3) We demonstrate a general way to integrate DFMs into various backbone networks to form DFNets.

Extensive comparison results on four datasets validate that the proposed DFM can consistently improve the classification performance and is superior to the state-of-the-art decision fusion-based methods [23], [24]. In addition, we validate the performance improvement on two relevant image recognition tasks, i.e., object detection and semantic segmentation, demonstrating the wide applicability of DFMs. Codes will be made public upon paper acceptance.

II. Related Work

A. Image Recognition With CNNs and Context Fusion

Advances in deep networks have greatly promoted image recognition. Since AlexNet [16] achieved a breakthrough on ImageNet [25], various networks have been proposed. VGG [17] and inception [26] demonstrated the benefits of increasing depth and multiple-scale receptive fields. ResNet [3], [27] enabled the training of deeper networks through the use of simple identity skip connections, and wide ResNet [28] showed the effectiveness of increasing the width of deep networks. Instead of simply deepening and widening networks, ShuffleNet [29] and MobileNet [30], [31] utilized pointwise group convolution, channel shuffling, and the novel inverted residual structure to greatly reduce computational costs while maintaining accuracy. Recently, with the increase in computing power, researchers have begun to use neural architecture search (NAS) to design new architectures, e.g., EfficientNet [32], significantly improving classification performance. While most of the above-mentioned methods focus on enhancing performance by increasing the complexity of deep networks or designing novel computationally efficient operations, we focus on designing an add-on module to enhance existing network architectures.

Besides network engineering, another commonly adopted strategy for improving image recognition is to fuse contextual information, e.g., spatial/channel context [33], [34], local/global context [35], and user-provided tags from extra community-contributed images [36], which describes the semantic content of images to some extent. We also aim to fuse contextual information to improve image recognition. Differently, the context fused by DFM is the category-coherent decisions.

B. Category Hierarchy

The arrangement of categories into a semantic hierarchy with many levels of abstraction has been well exploited [37]–[39]. Deng *et al.* [40] introduced hierarchy and exclusion graphs that capture the semantic relations between any two labels to improve classification performance. Yan *et al.* [41] proposed a two-level hierarchical CNN with the first layer separating easy classes using a coarse category classifier and the second layer handling difficult classes. To mimic the high-level reasoning ability of humans, Goo *et al.* [42] introduced a regulation layer that can abstract and differentiate object categories based on a given taxonomy, significantly improving the performance. However, a human-specified category hierarchy is not always available and may be not a good division from the perspective of CNNs.

C. Deep Decision Trees/Forests

Decision trees [43] are a major topic in the theory of belief functions theory [44]–[46] and have been well explored and utilized in the literature [47]. With the rise of deep networks, researchers have attempted to design deep decision trees or forests [48] to solve classification problems. To avoid operating in a greedy and local manner as in prevailing approaches that hinder representation training with CNNs, Kontschieder *et al.* [21] introduced a novel stochastic routing scheme for decision trees, enabling split node parameter learning via backpropagation. Without requiring the user to set the number of trees, Murthy *et al.* [24] proposed a "data-driven" deep decision network (DDN), in which decision stumps are introduced in a stagewise manner to classify samples with high confidence and partition the remaining data, which are more difficult to classify, into smaller data clusters for training successive expert networks in the next stage. Ahmed *et al.* [49] further proposed combining the training of a generalist model to discriminate coarse groupings of categories with the training of expert models aimed at accurate

recognition of classes within each grouping and obtained substantial improvement. Instead of clustering data based on image labels, Chen *et al.* [50] proposed a large-scale unsupervised maximum margin clustering technique to iteratively split images into a number of hierarchical clusters to train cluster-level CNNs at parent nodes and category-level CNNs at leaf nodes.

Instead of implementing each decision branch with another separate routing network, Xiong *et al.* [51] proposed a conditional CNN framework with dynamic routing through the activation of subsets of kernels, making the deep decision tree more compact. Based on this work, Baek *et al.* [23] proposed a fully connected (FC) "soft" decision jungle structure to enable the decision to be recoverable, and thus, lead to more discriminative intermediate representations and higher accuracies.

Our DFM can be regarded as a deep decision tree-based approach, and the most similar work to ours is [23]. However, the differences are at least threefold. First, instead of dynamically activating a subset of kernels to reduce the number of parameters, which would cause each kernel to be applied for only a subset of the decisions, our DFM, which adopts conditional codes to propagate decisions, can force each kernel to be applied for all decisions, and thus, make full use of the neurons. Second, their approach requires layers in which the number of channels is greater than the number of categories, which is a difficult requirement to satisfy in real cases with 1000 or more categories, while our solution does not have such restrictions. Finally, we design three novel loss functions to force DFM to make category-coherent decisions.

D. Information Propagation in CNNs

Information propagation has been widely exploited in various deep networks. Highway networks [52] allow information to propagate unimpeded across several layers on information highways. ResNets [27] propagate identities via well-defined residual block structures. More recently, DenseNets [18] have emerged, in which each convolution block is further enabled to receive raw information from all previous blocks. Compared with these skip connection-based architectures, which propagate identity feature maps, the decisions propagated in our approach are of lower dimensionality but provide more explicit (category-coherent) guidance. In addition, skip connections propagate feature maps to other layers over long distances to alleviate the degradation problem, while the decision fusion module (DFM) is a plug-in unit for refining the feature extraction process in subsequent layers. Furthermore, we will show that DFMs can be easily integrated into skip connection-based networks to further improve their performance.

E. Intermediate Supervision/Operation in CNNs

Intermediate supervision is a powerful regularization tool for training deep networks. In [26], [53], and [54], a few intermediate layers were connected to auxiliary classifiers mainly to address the problem of vanishing/exploding gradients. With the development of normalization techniques, e.g., intermediate normalization layers [55] and normalized initialization [56], the above-mentioned issues have been largely resolved. Our approach also adopts intermediate supervision. However, instead of regularizing the features in each intermediate layer, our supervision strategy is to force Decision Fusion Module to make category-coherent decisions. In addition, we apply indirect categorical supervision by determining categories at higher hierarchical levels, thus providing the intermediate layers with more flexibility in encoding "appropriate-level" features.

Another related work to ours is the SENet [57], in which the SE modules exploit the channel-aware attention mechanism to recalibrate intermediate features. Although with similar architectures, their differences from ours are at least five folds. First, the outputs of the SE modules represent the relative importance of different channels, while those of DFMs are category-coherent decisions. Second, the output channel dimensions of the SE modules are the same as the feature maps, while those of the decisions generated by the DFMs are coarse category numbers and are usually set as two. Third, the SE modules multiply the channel importance values with the feature maps, while the decisions generated by the DFMs are concatenated with them. Forth, the purpose of the SE modules is to recalibrate channelwise feature responses in current layers, while that of the DFMs is to make intermediate decisions to guide subsequent network layers. Fifth, the SE modules are trained without additional supervision, while DFMs require three extra intentionally designed loss functions to enforce the outputted decisions to be category coherent. Based on the above-mentioned differences, we conclude that our devised DFM is explored in an orthogonal way to the SE module. Moreover, we validate that these two modules can be combined together to obtain further improvement in Section V.

III. Decision Fusion Module

In this section, we first give the definition of category-coherent decisions and then introduce the structure of *DFM* together with three corresponding loss functions for supervision. Finally, we discuss the large category issue, which hinders DFM training and our solution.

A. Category-Coherent Decisions

Given inputs belonging to the same object category, if their corresponding decisions are similar, then these decisions can be called **category-coherent decisions**. Note that we also allow inputs belonging to multiple different categories to have similar decision results. In this article, we define the category-coherent decision in terms of n ($n \leq N$) **auxiliary categories**, namely, $\boldsymbol{D} = (D_1, \ldots, D_n)$, where $D_1 + D_2 + \cdots + D_n = 1$.

B. Structure of Decision Fusion Module

Decision Fusion Module (DFM) is a computational unit that is devised to make a category-coherent decision, *which determines the category at a higher hierarchical level*, based on the features encoded in the current layer and then combine the results with the original features before passing them to the subsequent network layers to guide them. A diagram of the DFM is shown in Fig. 1.

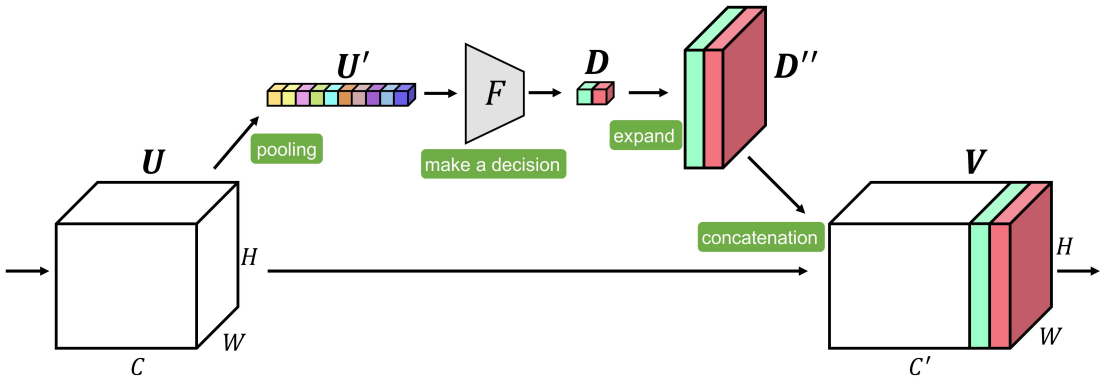


Fig. 1. Demonstration of Decision Fusion Module: given a feature map $\boldsymbol{U}$, we first conduct GAP to obtain the pooled feature map $\boldsymbol{U}'$ and then classify it using the network F to obtain an immediate decision $\boldsymbol{D}$. Finally, the decision is expanded and concatenated with the original feature map $\boldsymbol{U}$ to obtain a decision-combined feature map $\boldsymbol{V}$. Note that we propagate $\boldsymbol{V}$ instead of $\boldsymbol{U}$ to subsequent layers.

1) Making a Soft Decision: Given an intermediate feature map $\boldsymbol{U}$ ($\boldsymbol{U} \in \mathbb{R}^{C\times H\times W}$), our aim is to make a category-coherent decision to guide subsequent network layers without bringing too much additional computational cost. Therefore, instead of convolving on it continuously, we propose to adopt global average pooling (GAP) to greatly reduce the feature dimensions. As verified in [58], a pooled feature map $\boldsymbol{U}'$ with channelwise statistics is usually sufficiently discriminative for classification. We thus adopt an FC network F with one or two layers to make a decision on this basis. To facilitate the optimization of this decision branch along with the whole network structure in an end-to-end manner, we apply the Softmax function to the output, and thus, obtain a "soft" decision $\boldsymbol{D}$. In short, this decision is computed as

$$\boldsymbol{D} = \text{Softmax}(F(\text{GAP}(\boldsymbol{U}))) = \text{Softmax}(F(\boldsymbol{U}')). \tag{1}$$

2) Decision Fusion: Instead of dynamic routing based on the decision $\boldsymbol{D}$, as in deep decision trees [24], we treat $\boldsymbol{D}$ as a conditional code [22], on which the feature extraction process of later layers can be conditioned, to provide guidance. Specifically, we expand the decision vector $\boldsymbol{D}$ ($\boldsymbol{D} \in \mathbb{R}^{n\times 1}$) to have the same resolution as the feature map $\boldsymbol{U}$ by directly copying the decision scores and then concatenate the expanded decision $\boldsymbol{D}''$ ($\mathbf{D}'' \in \mathbb{R}^{n\times H\times W}$) with $\boldsymbol{U}$ as additional channels (see Fig. 1). In short, the decision fusion process is computed as follows:

$$\boldsymbol{V} = \text{Concat}(\boldsymbol{U}, \text{Expand}(\boldsymbol{D})) = \text{Concat}(\boldsymbol{U}, \boldsymbol{D}'') \tag{2}$$

where $\boldsymbol{V} \in \mathbb{R}^{C'\times H\times W}$ and $C' = C + n$.

C. Loss Functions for Supervising DFM

Enforcing DFM to make category-coherent decisions is challenging. Therefore, we intentionally design three loss functions to supervise DFM which are applied to its outputted decisions. In the following, we will describe them one by one.

1) Notations: We use $\mathcal{D}$ ($\mathcal{D} \in \mathbb{R}^{b\times n}$) to denote all the $\boldsymbol{D}$s in a mini-batch of size b, where $\mathcal{D}_{kj}$ is the decision score (confidence) of the jth auxiliary category for the kth instance in the batch.

2) Decision Explicit Loss: If the decision made by DFM is ambiguous, the subsequent layers will be unable to obtain much useful information from it. Therefore, to avoid all auxiliary categories having similar scores, we introduce a decision explicit loss to encourage the decision score to have a relatively larger value for one or several auxiliary categories. This loss function is defined as follows:

$$L_{\text{expl}}(\mathcal{D}) = \frac{1}{b}\sum_{k\in\{1,\ldots,b\}}\left(-\sum_{j\in\{1,\ldots,n\}}\mathcal{D}_{kj}\log\mathcal{D}_{kj}\right) \tag{3}$$

which has the form of entropy to encourage the decision scores for different auxiliary categories to vary widely.

3) Decision Consistent Loss: Simply enforcing the decision to be explicit is not enough. In addition, we desire the decisions made for many different instances with the same original category to be consistent. Specifically, their decision scores for the same auxiliary category should be similar. Therefore, we propose a decision consistent loss, which is defined as follows:

$$L_{\text{cons}}(\mathcal{D}) = \frac{1}{Nn}\sum_{i\in\{1,\ldots,N\}}\sum_{j\in\{1,\ldots,n\}}\mathcal{V}_{ij} \tag{4}$$

where $\mathcal{V}_{ij}$ is the sample variance of $\mathcal{D}_{kj}$ ($k \in \{1,\ldots,b\}$) for the instances in the batch whose original category is i.

Let $\mathcal{I}$ ($\mathcal{I} \in \mathbb{R}^{b\times N}$) denote an indicator matrix for a batch of data: if the original category of the kth instance in the batch is i, then $\mathcal{I}_{ki} = 1$; otherwise, $\mathcal{I}_{ki} = 0$. Thus, the mean decision score $\mathcal{M}_{ij}$ for the jth auxiliary category for all the instances in the batch with original category i can be calculated using the following equation:

$$\mathcal{M}_{ij} = \frac{\sum_{k\in\{1,\ldots,b\}}\left(\mathcal{I}_{ki}\mathcal{D}_{kj}\right)}{\sum_{k\in\{1,\ldots,b\}}\mathcal{I}_{ki} + \delta} \tag{5}$$

where δ is a small value introduced to avoid the divide-by-zero error. Then, we can calculate $\mathcal{V}_{ij}$ as

$$\mathcal{V}_{ij} = \frac{\sum_{k\in\{1,\ldots,b\}}\mathcal{I}_{ki}\left(\mathcal{D}_{kj} - \mathcal{M}_{ij}\right)^2}{\sum_{k\in\{1,\ldots,b\}}\mathcal{I}_{ki} - 1 + \delta}. \tag{6}$$

By substituting (5) into (6) and expanding the formulation, we obtain a new equation

$$\mathcal{V}_{ij} = \frac{\sum_{k\in\{1,\dots,b\}} \mathcal{I}_{ki}\mathcal{D}_{kj}{}^2}{\sum_{k\in\{1,\dots,b\}} \mathcal{I}_{ki} - 1 + \delta} - \frac{\left(\sum_{k\in\{1,\dots,b\}} \mathcal{I}_{ki}\mathcal{D}_{kj}\right)^2}{\left(\sum_{k\in\{1,\dots,b\}} \mathcal{I}_{ki} + \delta\right)\left(\sum_{k\in\{1,\dots,b\}} \mathcal{I}_{ki} - 1 + \delta\right)}. \quad (7)$$

However, calculating the $\mathcal{V}_{ij}$ one by one would be very time-consuming. Therefore, we leverage matrix operations to accelerate the calculation. The derived equation is as follows:

$$\mathcal{V} = \frac{\mathcal{I}^{\mathrm{T}} \times (\mathcal{D}\mathcal{D})}{\mathcal{I}' - 1} - \frac{\left(\mathcal{I}^{\mathrm{T}} \times \mathcal{D}\right)\left(\mathcal{I}^{\mathrm{T}} \times \mathcal{D}\right)}{\mathcal{I}'(\mathcal{I}' - 1)} \quad (8)$$

which is a matrix where the value in the ith row and jth column is $\mathcal{V}_{ij}$, meaning that $\mathcal{V} \in \mathbb{R}^{N\times n}$. The operator $\times$ represents the cross product, while all other operations are performed elementwisely. $\mathcal{I}' \in \mathbb{R}^{N\times n}$ is a 2-D matrix with $\mathcal{I}'_{ij} = \sum_{k\in\{1,\dots,b\}} \mathcal{I}_{ki} + \delta$ for arbitrary $j \in \{1, \dots, n\}$. Although simple, it is critical to calculate the loss in an efficient mode.

4) Decision Balance Loss: To avoid the degenerate situation in which DFM explicitly assigns all input instances with different original categories to a single-auxiliary category, we introduce a decision balance loss. Ideally, we expect all auxiliary categories to be well balanced when assigned to instances belonging to all the original categories. Due to the limited batch size, we simply formulate this loss as the inverse of entropy using all instances in a mini-batch

$$L_{\mathrm{bal}}(\mathcal{D}) = \sum_{j\in\{1,\dots,n\}} m_j \log m_j$$
$$m_j = \sum_{k\in\{1,\dots,b\}} \mathcal{D}_{kj} + \delta. \quad (9)$$

D. Large Category Issue

For all the original categories that appear in a batch, we expect their decision scores to be consistently and explicitly distributed in a balanced manner among all the auxiliary categories. However, due to limited computational resources and the challenges of large-batch SGD training [59], the batch size is typically set to a small number, e.g., 128. For tasks with 100, 1000, or more original categories, randomly loading a batch of data will result in the number of instances per batch that belong to each original category being only two or even smaller, making L_{cons} ineffective.

Therefore, instead of simply increasing the batch size to maintain more data samples in each batch, we propose a novel load-shuffle-split strategy that can resolve the large category issue without significantly enlarging the batch size. Specifically, consider the example shown in Fig. 2, where the number of original categories is 4 and the batch size is 4. The load-shuffle-split strategy has three key steps: 1) instead of loading only four data samples, we load more data samples in each iteration (e.g., eight in our example); 2) we first generate a number list $[1, 2, 3, 4]$ containing all the category IDs and then shuffle it to obtain another list $[1, 4, 3, 2]$, which is finally split into two lists, $[1, 4]$ and $[3, 2]$; and 3) we split the eight data samples into two training batches according to the category IDs in the two split lists generated in the previous step (see Fig. 2 again) and then use these two batches separately for training. In this way, the number of instances of each original category in a training batch is doubled. Note that the purpose of the load-shuffle-split strategy is simply to ensure that the number of samples belonging to each original category in a single batch is sufficiently large for better estimation of L_{cons}, and it does not require the numbers of samples belonging to different original categories in a batch to be the same.

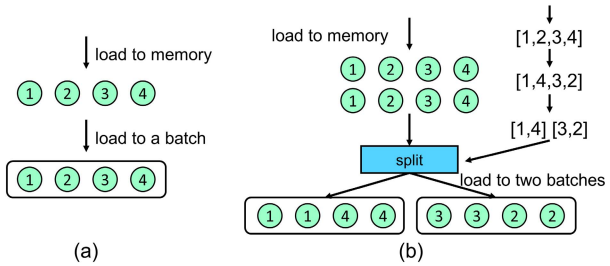


Fig. 2. Demonstrations of loading data belonging to four categories (a circle with a digit i inside indicates an instance whose original category is i) into batches of size 4: (a) in the traditional approach, the number of instances of each category in a training batch will be 1 and (b) in our approach, more data are loaded into memory at once, and the samples are shuffled and then split into multiple batches, thereby increasing the possible number of instances of the same category in each batch.

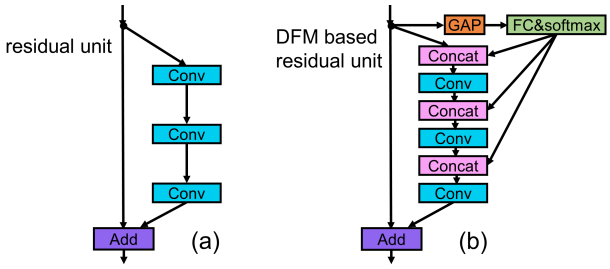


Fig. 3. Schematics of (a) residual unit and (b) DFM-based residual unit.

IV. Decision Fusion Networks

DFMs are flexible and can be integrated into various classification network architectures to form DFNets. As it is straightforward to apply them to VGG network [17] or AlexNet [16], in this section, we illustrate how to integrate DFMs into several sophisticated modern architectures. Finally, we describe how to train DFNets.

A. Exemplar Decision Fusion Networks

For residual networks, e.g., ResNet [3], since they are organized by stacking residual blocks, we integrate DFM into each of their residual units. Specifically, intermediate category-coherent decisions are incorporated along the residual branch, and thus, these neurons can be guided to learn better residuals, see Fig. 3. For the inception module, we demonstrate integrating DFM into the beginning of all the branches, such that the neurons for handling different receptive fields can be guided, see Fig. 4. The integration of DFMs with many

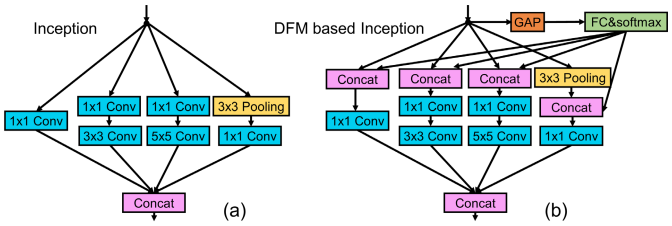


Fig. 4. Schematics of (a) inception module and (b) DFM-based inception module.

other ResNet and InceptionNet variants, such as ResNeXt [60] and inception-ResNet [61], and other popular architectures could be achieved using similar schemes. Note that we only demonstrate one possible way to integrate DFM, and the integration can be implemented in different ways.

B. Training of Decision Fusion Networks

The training of DFNets for image classification is similar to the training of their backbone networks, although with additional losses for DFMs. Therefore, the final loss function of DFNets is the sum of the traditional cross-entropy loss, which is used for classification, and the three losses for supervising DFMs. In short, the final loss function is computed as follows:

$$L = L_{\text{CE}} + \frac{1}{|\boldsymbol{D}|}\sum_{\mathcal{D}\in\boldsymbol{D}}\left(\alpha_1 L_{\text{expl}}(\mathcal{D}) + \alpha_2 L_{\text{cons}}(\mathcal{D}) + \alpha_3 L_{\text{bal}}(\mathcal{D})\right) \tag{10}$$

where L_{CE} is the cross-entropy loss, $\boldsymbol{D}$ denotes all the decisions made by DFMs in DFNets, $|\boldsymbol{D}|$ is the number of decisions, and α_1, α_2, and α_3 are weighting parameters. Note that each DFM makes only one decision, and there are multiple DFMs in the DFNets.

V. Experimental Results

In this section, we evaluate the image classification performance of our approach on four publicly available datasets. Our main focus is *to demonstrate that DFMs can improve the performance of backbone CNN networks for image classification*, rather than to challenge the records of current state-of-the-art approaches. Therefore, we spend more space comparing our approach with popular baseline networks on three relatively small-scale datasets due to limited computational resources, and we then report the results of our approach on the ImageNet 2012 dataset [25] to validate its scalability. Finally, we extend our method to handle two relevant image recognition applications to demonstrate the generalization ability.

A. Implementation

We implement DFM and reproduce all the evaluated networks with PyTorch [62]. The decision network F of DFM consists of two FC layers surrounding an ReLU layer [63], followed by a Softmax layer to normalize the decision scores. To limit the model complexity, we reduce the dimensions in the first FC layer with a reduction ratio of 16. For all DFMs integrated into a network, we assume that their numbers of auxiliary categories are exactly the same, and we set this number to 2 if not specifically stated. For VGG, we add a batch normalization (BN) layer [55] with no dropout [64] and use one FC layer. For Inception, we choose v1 with BN. The other models are identical to those presented in the original papers. Note that the FC layers in DFM could be implemented with the efficient 1×1 convolutional operators. To provide a fair complexity comparison with SENets, we still report the statistics of codes with FC layers.

B. Datasets and Training Details

1) CIFAR-10 and CIFAR-100 [65]: These datasets consist of 60 k 32×32 images that belong to ten and 100 categories, respectively. We train the models on the whole training set of 50 k images with a mini-batch size of 128 and evaluate them on the test set. We set the initial learning rate as 0.1, and divide it by 5 at 60, 120, and 160 epochs for a total of 200 epochs. For data augmentation, we pad four pixels on each side of the image, randomly sample a 32×32 crop from the padded image or its horizontal flipped version, and then apply the simple mean/std normalization.

2) CINIC-10 [66]: This dataset contains 270 k 32×32 images belonging to ten categories, equally split into three subsets for training, validation, and testing. We train the models on the training set with a batch size of 128 and evaluate them on the test set. Training starts with an initial learning rate of 0.1 and is cosine annealed to zero over a total of 300 epochs, and the same data augmentation scheme used for the CIFAR datasets is applied.

3) ImageNet [25]: This dataset consists of 1.2 million training images and 50 k validation images from 1 k classes. We train the models with minimal data augmentation, including random resized cropping, flipping, and mean/std normalization, on the training set, and we report the results obtained on the validation set. The learning rate is initially set to 0.1 and decreased by a factor of 10 every 30 epochs over a total of 90 epochs.

During training, the three loss functions are calculated for all DFMs in each evaluated model, and the average of each loss is accumulated with the traditional cross-entropy loss for classification. We set the weights of the three loss terms as 0.01 on ImageNet and 0.1 on the other datasets via cross-validation. All models are trained from scratch with SGD as the optimizer using the default parameters, and the weights are initialized following [56]. The final reported results are averages based on three independent runs. For each run, we evaluate the single-crop performance in each epoch and choose the best one.

C. CIFAR and CINIC-10 Experiments

To evaluate the effectiveness of DFM, we first conduct extensive ablation studies on three relatively small datasets to verify that DFNets outperform the corresponding baseline networks without bells and whistles and then compare them with the state-of-the-art methods to demonstrate the superiority.

TABLE I

CLASSIFICATION RESULTS ON CIFAR-100 WITH DIFFERENT NUMBERS OF CATEGORIES IN A BATCH. "#IMAGES IN AN ITER." REFERS TO THE NUMBER OF IMAGES LOADED INTO MEMORY IN EACH ITERATION, AND "#CATEGORIES IN A BATCH" REFERS TO THE NUMBER OF CATEGORIES APPEARING IN A BATCH. THE RESULTS DENOTED BY "-" ARE OBTAINED WITHOUT APPLYING THE LOAD-SHUFFLE-SPLIT STRATEGY

Architecture	#Categories in a batch	#Images in an iter.	Acc (%)	
			Top-1	Top-5
ResNet-20	-	-	68.82	91.03
ResNet-20	25	512	68.93	90.98
ResNet-20	10	1280	64.88	89.54
DF-ResNet-20	-	-	69.50	91.69
DF-ResNet-20	50	256	69.80	91.98
DF-ResNet-20	25	512	**70.51**	**92.25**
DF-ResNet-20	10	1280	65.34	89.87
ResNet-56	-	-	72.23	92.36
ResNet-56	25	512	72.17	92.41
ResNet-56	10	1280	54.13	78.65
DF-ResNet-56	-	-	73.41	93.18
DF-ResNet-56	50	256	73.58	93.24
DF-ResNet-56	25	512	**73.76**	**93.39**
DF-ResNet-56	10	1280	55.73	79.86

1) Number of Categories in Each Batch and Load-Shuffle-Split Strategy: As mentioned earlier, we proposed the load-shuffle-split strategy to handle the large category issue, which affects the estimation of decision consistent loss. In this section, we evaluate the effects of the proposed strategy. Since CIFAR-100 consists of images belonging to 100 categories, while the mini-batch size is 128, we choose this dataset for this evaluation. Specifically, we consider four different configurations, each with approximately 128 images per training batch to ensure fair comparisons. The results in Table I show that DF-ResNet-20 and DF-ResNet-56 obtain ∼0.8% improvement in the top-1 accuracy even without applying the load-shuffle-split strategy. After applying the strategy to force that each training batch should contain images belonging to 25 categories, the improvement in the top-1 accuracy is further enlarged to ∼1.7%. Therefore, we conclude that our load-shuffle-split strategy is useful for training DFNets.

However, we would like to emphasize that *the improvement is not achieved because of the load-shuffle-split strategy but because of the better estimation of* L_{cons} when training DFNets. Indeed, the load-shuffle-split strategy itself is useless and even harmful for classification since it distorts the distribution of the data samples, which are required to be independent and identically distributed. Good evidence is reported in Table I showing that the performance of ResNet-20 and ResNet-56 with 25 categories in each batch is comparable to that of the baselines. However, when the number of categories per training batch decreases to 10, we can see that the performance of DF-ResNets drops sharply, while that of the baseline ResNet-20 and ResNet-56 architectures drops even heavier.

2) Number of Auxiliary Categories: We investigate the effect of the number of auxiliary categories used in DFM with the number of original categories existing in each training batch set to 25 based on the above-mentioned experiments, and we report the experimental results in Table II. It can be seen that the configuration with two auxiliary categories brings consistent improvements, while those configurations with larger numbers of auxiliary categories are unstable. The reason is probably that the current supervision is not sufficient to force DFM to make use of more auxiliary categories, since there is no ground truth for setting the correspondence between the original and auxiliary categories. Therefore, we simply set the number of auxiliary categories to 2. In any case, in Section V-C.3, we will show that it is the decision scores that represent the categories at a higher hierarchical level, rather than the auxiliary categories themselves.

TABLE II

CLASSIFICATION RESULTS ON CIFAR-100 WITH DIFFERENT NUMBERS OF AUXILIARY CATEGORIES FOR INTERMEDIATE DECISIONS AND DECISION FUSION. NOTE THAT THE RESULTS FOR AUXILIARY CATEGORY = 100 CORRESPOND TO THE USE OF THE ACTUAL CATEGORIES TO SUPERVISE THE DECISIONS AND A ✓ MARK IN THE "FUSION" COLUMN INDICATES THAT THE DECISIONS ARE FUSED WITH THE FEATURES, FF- INDICATES ADOPTING FEATURE FUSION INSTEAD OF DECISION FUSION

Architecture	#Aux. categories / channels	Fusion	Acc (%)	
			Top-1	Top-5
ResNet-20	-		68.82	91.03
DF-ResNet-20	2		69.15	91.32
DF-ResNet-20	5		69.11	91.68
DF-ResNet-20	10		69.55	91.42
DF-ResNet-20	25		69.31	91.24
DF-ResNet-20	100		69.09	91.18
DF-ResNet-20	2	✓	**70.51**	**92.52**
DF-ResNet-20	5	✓	69.49	91.65
DF-ResNet-20	10	✓	69.91	91.56
DF-ResNet-20	25	✓	69.41	91.67
DF-ResNet-20	100	✓	68.81	91.09
FF-ResNet-20	2	✓	69.16	91.55
FF-ResNet-20	5	✓	69.14	91.77
FF-ResNet-20	10	✓	69.43	92.05
ResNet-56	-		72.23	92.36
DF-ResNet-56	2		73.29	93.01
DF-ResNet-56	5		72.37	92.87
DF-ResNet-56	10		72.93	93.08
DF-ResNet-56	25		73.09	92.94
DF-ResNet-56	100		72.58	92.69
DF-ResNet-56	2	✓	73.76	**93.39**
DF-ResNet-56	5	✓	**73.86**	93.28
DF-ResNet-56	10	✓	73.51	93.32
DF-ResNet-56	25	✓	73.02	92.95
DF-ResNet-56	100	✓	71.80	92.37
FF-ResNet-56	2	✓	72.15	92.67
FF-ResNet-56	5	✓	72.45	92.90
FF-ResNet-56	10	✓	72.93	92.89

3) Actual Categories: We investigate the use of the actual categories to supervise the decisions on CIFAR-100 by setting the number of auxiliary categories to 100 and applying the cross-entropy loss to the decisions made by DFMs. The results in Table II show that DF-ResNets with decision fusion supervised by the actual categories do not bring any improvement. The reasons are perhaps twofold: 1) the number of actual categories is much larger than the number of auxiliary categories, which represent categories at a higher hierarchical level, and thus, it is more challenging to make accurate decisions and 2) forcing early layers to classify the actual categories will hinder them from learning low-level features, and thus, will eventually affect the ability of the later layers to abstract high-level features from low-level features.

4) Intermediate Supervision and Decision Fusion: We investigate whether the improvement could be achieved

TABLE III
CLASSIFICATION RESULTS OF DF-RESNETS ON THE CIFAR-100 DATASET WITH PARTIAL ABLATION OF THE LOSS FUNCTIONS

Architecture	L_{expl}	L_{cons}	L_{bal}	Acc (%)	
				Top-1	Top-5
ResNet-20				68.82	91.03
DF-ResNet-20				69.07	91.09
DF-ResNet-20		✓	✓	70.31	92.11
DF-ResNet-20	✓		✓	70.07	92.05
DF-ResNet-20	✓	✓		69.44	91.88
DF-ResNet-20	✓	✓	✓	**70.51**	**92.25**
ResNet-56				72.23	92.36
DF-ResNet-56				72.28	92.31
DF-ResNet-56		✓	✓	73.27	93.19
DF-ResNet-56	✓		✓	73.52	93.15
DF-ResNet-56	✓	✓		72.87	92.73
DF-ResNet-56	✓	✓	✓	**73.76**	**93.39**

through the intermediate supervision of the auxiliary categories alone without decision fusion. The results in Table II show that all DF-ResNet-20 and DF-ResNet-56 models without decision fusion exhibit better performance than their corresponding baselines, validating the usefulness of intermediate supervision for deep networks. Similar to the results with decision fusion, the networks without decision fusion perform the best when using two auxiliary categories. In particular, the models supervised using the actual categories, as in the case of deeply supervised nets [53] perform the worst, validating that auxiliary categories offer more flexibility for intermediate supervision than the actual categories.

By comparing models with the same number of auxiliary categories in Table II that do and do not adopt decision fusion, we can see that both the DF-ResNet-20 and DF-ResNet-56 models with decision fusion perform better, validating the usefulness of decision fusion. In contrast, perhaps because the decisions supervised by the actual categories are already unstable, their results with decision fusion are worse.

5) Decision Fusion Versus Feature Fusion: We investigate whether decision fusion is better than feature fusion with similar numbers of parameters. Specifically, we replace DFM with 1×1 convolutional layers to project the original feature maps instead of the pooled ones into compressed feature maps with channel sizes of 2, 5, and 10, and then fuse them with the original feature maps. Since the FC layers in DFM could also be implemented with 1×1 convolutional operators, the numbers of parameters in the two methods are similar. The results in Table II show that ResNets with feature fusion can obtain slightly better performance than baselines, but are far behind our proposed approach that adopts decision fusion.

6) Importance of Three Loss Functions: We conduct extensive ablation experiments to investigate the influence of each loss function in training DFMs. The results summarized in Table III show that the performance drops if we remove any loss functions, validating that all three properties, i.e., explicit, consistent, and balanced, enforced by these losses are critical to the outputted decisions by DFMs to be category coherent for guiding subsequent network layers. In particular, L_{bal} has the largest influence on the classification results; without it, the top-1 accuracies drop by $\sim$1% for both models. It probably indicates that DFMs are much easier to be degenerated to assign all different input feature maps to one single auxiliary category, compared with the other two degenerated situations suppressed by the decision explicit loss and the decision consistent loss. In addition, we note that the DF-ResNets with one loss function ablated could still outperform the baseline networks, thereby validating the effectiveness of DFMs.

To further validate whether the improvement is attributable to the slightly enlarged architectures or the regularized decisions, we also report the results of DF-ResNets without any loss functions for supervising DFMs. The results in Table III show almost no improvement. Therefore, we conclude that the improvement is brought by the instructive decisions guided by the three loss functions.

7) Comparisons With the State-of-the-Art Methods: We conduct extensive experiments on three challenging datasets: CIFAR-10, CIFAR-100, and CINIC-10, with various popular architectures, including ResNet [3], network in network (NIN) [58], GoogLeNet [26], VGG [17], and DenseNets [18], as backbones. The results reported in Table IV show that all networks could achieve significantly better performance with the integration of DFMs (e.g., more than 1.5% top-1 accuracy improvement for DF-ResNet-56 on CIFAR-100), validating the effectiveness and versatility of DFMs. In particular, we would like to point out that DF-ResNet-56 outperforms the ResNet-110 on both CIFAR10 and CINIC-10 and is comparable to it on CIFAR-100, but with nearly half the numbers of parameters and multiply-and-accumulate operations (MACs). In addition, we choose to compare our approach with two recent decision tree-based methods: deep convolutional decision jungle (DCDJ) [23] and DDN [24]. Since the authors of these methods have not released their codes, we thus compare our results with those reported in their papers. It could be seen that DF-NIN outperforms DCDJ and DDN when using a NIN as the backbone network. Finally, we compare DFM with squeeze-and-excitation (SE) module [57], which also adopts GAP and FC layers to generate intermediate results. In Table IV, we can see that DFM is superior to or at least comparable to the SE module.

8) Complexity Analysis: For practical use, DFMs are expected to provide an effective tradeoff between complexity and performance. Therefore, we report complexity statistics in Table IV. With the integration of DFMs, the increases in the number of parameters and MACs are fewer than 5% of their original values. Meanwhile, in Section V-C7, we have validated that the brought improvements are significant. Therefore, we can conclude that the overhead introduced by DFM is acceptable.

9) Visualization and Discussion: To investigate what DFMs learn, we visualize the decisions made by the nine DFMs of DF-ResNet-20 on 512 images from the CIFAR-10 dataset, as shown in Fig. 5, where DFMs of the earliest layer to the last layer are arranged from left to right in sequence. For the decisions made by each DFM, all 512 images are visualized in positions related to the decision scores assigned to them: the larger the decision score assigned to the first auxiliary category (out of a total of two) is, the lower the position of the image, while all images are randomly spread in the horizontal direction. Since under weak supervision

TABLE IV

CLASSIFICATION RESULTS ON THE CIFAR-10, CIFAR-100, AND CINIC-10 DATASETS. NOTE THAT THE NUMBERS OF PARAMETERS AND MACS ARE CALCULATED BASED ON THE EXPERIMENTS ON CIFAR-10. THE NUMBERS IN BRACKETS DENOTE THE PERFORMANCE IMPROVEMENT. * INDICATES RESULTS THAT ARE REPORTED IN THE ORIGINAL PAPERS

Architecture	#Params	#MACs	Acc (%)		
			CIFAR10	CIFAR-100	CINIC-10
NIN [58]	996.99k	0.22G	89.71	67.76	80.10
DDN [24] *	-	-	90.32	68.35	-
DCDJ [23] *	-	-	-	68.80	-
DF-NIN	997.90k	0.23G	**90.87**(1.16)	**69.11**(1.35)	**81.07**(0.97)
ResNet-56 [3]	855.77k	0.13G	93.62	72.23	84.74
SE-ResNet-56 [57]	861.82k	0.13G	94.28	**73.81**	85.09
DF-ResNet-56	894.96k	0.14G	**94.35**(0.73)	73.76(1.53)	**85.50**(0.76)
ResNet-110 [3]	1.73M	0.26G	93.98	73.94	85.18
SE-ResNet-110 [57]	1.74M	0.26G	**94.58**	74.42	85.57
DF-ResNet-110	1.81M	0.28G	94.56(0.58)	**74.85**(0.91)	**86.34**(1.16)
GoogLeNet [26]	6.13M	1.53G	95.27	79.41	87.89
DF-GoogLeNet	6.35M	1.54G	**95.65**(0.38)	**80.73**(1.32)	**88.31**(0.42)
VGG13 [17]	9.42M	0.23G	94.18	74.42	85.04
DF-VGG13	9.49M	0.23G	**94.61**(0.43)	**74.94**(0.52)	**85.59**(0.55)
DenseNet-BC-40-12	0.17M	0.074G	92.62	70.61	83.80
DF-DenseNet-BC-40-12	0.19M	0.077G	**93.51**(0.89)	**72.40**(1.79)	**84.39**(0.59)
DenseNet-BC-100-12	0.77M	0.30G	95.08	77.25	87.08
DF-DenseNet-BC-100-12	0.89M	0.31G	**95.52**(0.44)	**78.32**(1.07)	**87.28**(0.20)

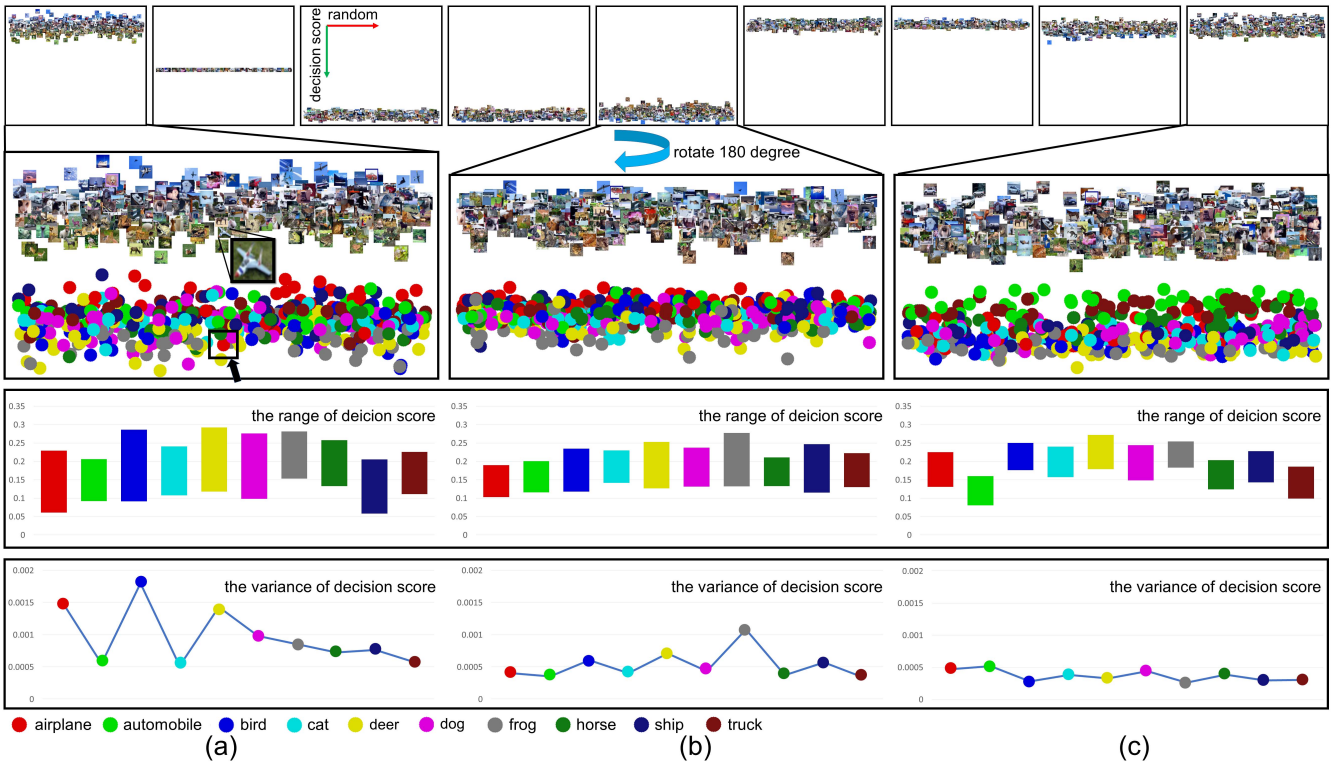


Fig. 5. **Row 1**: Each column visualizes the decisions made by one DFM (out of a total of nine) in DF-ResNet-20 on 512 images in CIFAR-10 (from left to right: DFMs of early layers to later layers). The vertical position of each image indicates the decision score for the first auxiliary category, namely, $\boldsymbol{D}[0]$ with a position closer to the bottom indicating a larger decision score, while the images are randomly spread in the horizontal direction. **Row 2**: Zoomed-in views of the decisions and the corresponding original categories as represented by colored circles. **Row 3**: The ranges of the decision scores generated by three DFM corresponding to the zoomed-in views above for the ten original categories. **Row 4**: The variances of the above-mentioned decision scores. Note that we visualize the decision scores of the fifth DFM as $1 - \boldsymbol{D}[0]$ for better comparison, and **(a)–(c)** are used to indicate the results of the first, fifth, and last DFMs, respectively.

applied by the decision balance loss that is estimated inaccurately, the decision scores are concentrated within a range of $[0.05, 0.3]$ instead of spanning the whole $[0, 1]$ range (see Row 3 of Fig. 5). Even though, these decisions are proved to be informative to guide later layers for obtaining better classification performance.

We can see that the images in the first column are distributed in the vertical direction with "blue" images located above,

TABLE V
CLASSIFICATION RESULTS ON IMAGENET. NOTE THAT THE RESULTS DENOTED BY "-" ARE OBTAINED WITHOUT APPLYING THE LOAD-SHUFFLE-SPLIT STRATEGY

Architecture	#Categories in a batch	Batch size	Acc (%)	
			Top-1	Top-5
ResNet-18	-	256	69.81	89.79
SE-ResNet-18	-	256	70.34	89.95
DF-ResNet-18	-	256	70.40	90.07
DF-ResNet-18	250	1024	**70.85**	**90.25**
SE-DF-ResNet-18	250	1024	**71.24**	**90.41**
ResNet-50	-	256	75.50	92.51
SE-ResNet-50	-	256	76.45	93.13
DF-ResNet-50	-	256	76.55	93.29
DF-ResNet-50	250	1024	**77.23**	**93.52**
SE-DF-ResNet-50	250	1024	**77.95**	**93.83**

Architecture	#Categories in a batch	Batch size	Acc (%)	
			Top-1	Top-5
GoogLeNet	-	256	70.68	90.08
SE-GoogLeNet	-	256	71.13	90.25
DF-GoogLeNet	-	256	71.22	90.37
DF-GoogLeNet	250	1024	**71.66**	**90.54**
SE-DF-GoogLeNet	250	1024	**72.02**	**90.69**
ResNet-101	-	256	76.71	93.24
SE-ResNet-101	-	256	77.57	93.66
DF-ResNet-101	-	256	77.64	93.82
DF-ResNet-101	250	1024	**78.26**	**93.93**
SE-DF-ResNet-101	250	1024	**78.86**	**94.17**

and "green" images below, indicating that the first DFM probably makes its decisions based on color. Although this is a simple division, frogs, and airplanes are separated quite well, validating that such low-level information is useful for classification. The second DFM seems to be ineffective, assigning equal decision scores to the two auxiliary categories. This behavior is similar to that of the ResNet, which allows some gated shortcuts to be closed, indicating that DFM could also be closed automatically to resist overfitting. Interestingly, the third–fifth DFMs make almost inverted decisions relative to the others (i.e., the scores they generate are approximately one minus the scores generated by the other DFMs), indicating that the auxiliary categories constructed by different DFMs could be different, and neural networks have the ability to decode these decisions. By rotating the decisions in the fifth column and visualizing the scores as $1-\boldsymbol{D}[0]$ instead, we find that these decisions are somewhat consistent with those in the first column, but exhibit better semantic clustering along the vertical direction [see the ranges and variances of the decision scores in Fig. 5(a) and (b)]. Note that a "green" airplane is originally located in the lower region by mistake in the first set of decisions [see the black rectangle in Row 2 of Fig. 5(a)], while all airplanes [see the red circles in Row 2 of Fig. 5(b)] are located in the upper region in the latter set of decision, demonstrating that *our approach can recover from some false decisions made previously.* From the visualization of the decisions made by the last DFM, we find that the instances in the same categories are located within a small range along the vertical axis and are well separated from some other categories [see the ranges and variances of the decision scores in Fig. 5(c)]. Therefore, we conclude that *it is the decision score that encodes some meaningful information about the object category, rather than the auxiliary category itself.* According to the ranges and variances of the decision scores reported in Rows 3 and 4 of Fig. 5, we can observe that the decisions are progressively improved, validating our intuition that the features and decisions are iteratively refined. In particular, trucks and automobiles, which can both be considered to belong to the broader category "transportation" in the category hierarchy, are located in similar vertical ranges [see Row 3 of Fig. 5(c)]. However, some other "transportation" objects (e.g., ships), are mixed with objects belonging to the coarse category "animals" (e.g., horses). Therefore, we conclude that the decisions made by DFM are not necessarily consistent with a human-specified category hierarchy but rather follow another division scheme that is better from the perspective of CNNs.

D. ImageNet Experiments

To evaluate the effectiveness of DFM on large-scale datasets, we apply it in combination with common networks, lightweight networks, and NAS [68] networks on ImageNet [25] and then visualize the resulting feature maps to obtain a better understanding of how DFM works.

1) Common Networks: We evaluate the performance of various DFNets with common networks as backbones on ImageNet. The results reported in Table V show that the DFNets outperform all of the baseline networks by large margins (i.e., $\sim$0.6% for ResNet-18 and GoogLeNet and $\sim$1.0% for ResNet-50 and ResNet-101) when 1000-category instances are randomly sampled into batches with a size of 256, in which case L_{cons} can contribute little to the training process. Therefore, we also conduct experiments with the batch size set to 1024, and the number of categories in a batch enforced to be 250 using the load-shuffle-split strategy. We can see that the improvements in the top-1 accuracy for all DFNets are finally enlarged to 1.0%–1.6%, validating the scalability of our method.

We also compare our method with SENets. The results in Table V show that DFNets outperform the corresponding SENets for all four backbone networks. In addition, all DFNets could be further improved by adding the SE module, validating the complementarity of our method with respect to SENets.

2) Lightweight Networks: Since the overhead introduced by DFM is quite small in terms of both parameters and computation, we investigate the incorporation of DFMs into the lightweight network, MobileNetV3 [69]. We adopt the same training settings as in the other ImageNet experiments in this work, except that we set the weight decay to 4e-5 instead of 1e-4 and use a cosine-decay learning rate policy (decreased from 0.05 to 0) for 150 epochs. The results in Table VI show that DF-MobileNetV3-small significantly outperforms MobileNetV3-small with minimal additional computational cost, validating the great potential of incorporating DFMs into applications running on low-end devices. In addition, we would like to point out that MobileNetV3-small already adopts SE blocks [57] to boost performance; thus, the further

TABLE VI

CLASSIFICATION RESULTS OF MOBILENETV3-SMALL AND DF-MOBILENETV3-SMALL ON IMAGENET. NOTE THAT #MACS DENOTES THE NUMBER OF MACS

Architecture	#Params	#MACs	Top-1 Acc(%)	Top-5 Acc(%)
MobileNetV3-small	3.11M	62.89M	67.10	86.37
DF-MobileNetV3-small	3.12M	65.24M	**68.08**	**86.71**

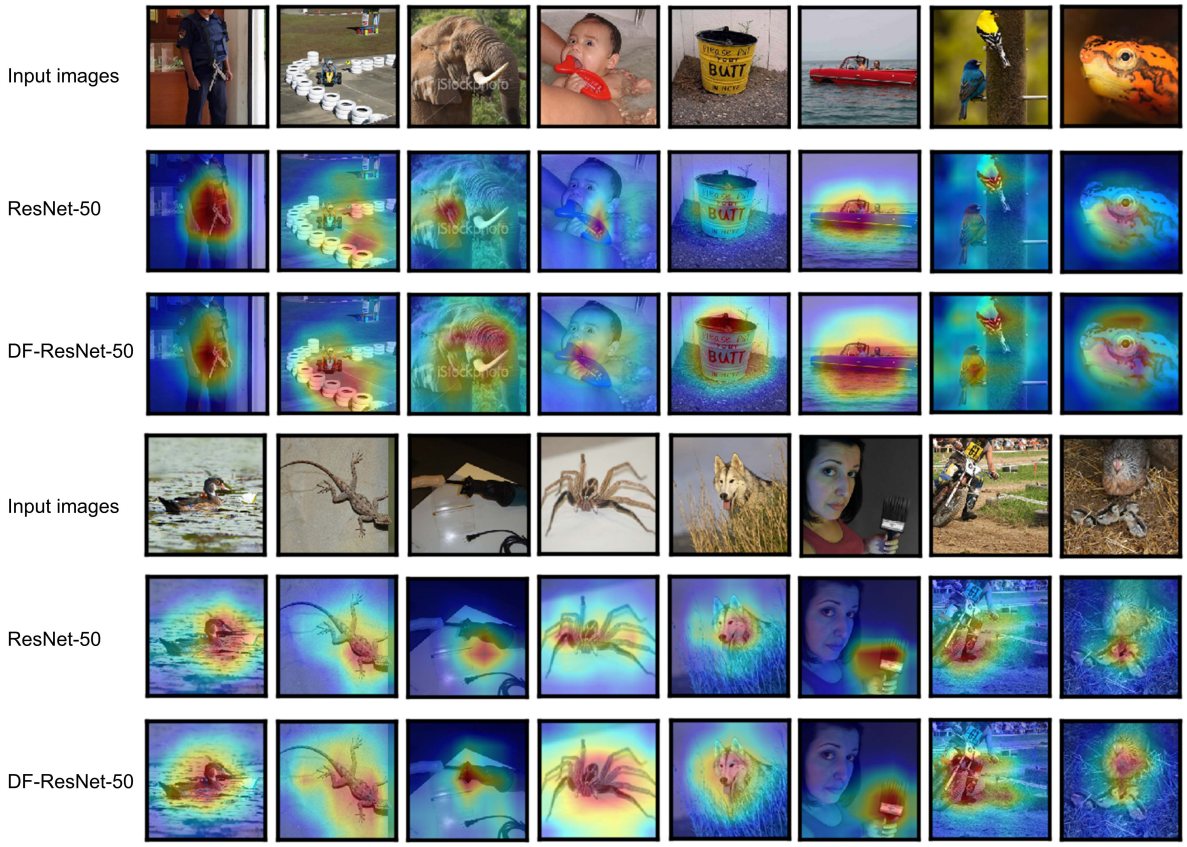


Fig. 6. Activation-based attention maps [67] based on the last residual group of the original ResNet-50 and DF-ResNet-50 architectures.

TABLE VII

CLASSIFICATION RESULTS OF EFFICIENTNETS AND DF-EFFICIENTNETS ON IMAGENET

	Top-1 Acc(%)	Top-5 Acc(%)
EfficientNet-B0	77.4	95.3
DF-EfficientNet-B0	**78.7**	**95.9**
EfficientNet-B2	80.4	95.1
DF-EfficientNet-B2	**81.6**	**95.6**
EfficientNet-B3	81.5	95.7
DF-EfficientNet-B3	**82.1**	**95.9**

increased accuracy of DF-MobileNetV3-small validates that DFM can be combined with many other modules (e.g., SE blocks) together to obtain even better results.

3) NAS Networks: Since NAS [68] methods have shown superior performance compared with manually designed architectures for image classification, we, thus, investigate incorporating DFMs into the state-of-the-art NAS network, EfficientNet [32]. We follow the same training strategy as for the open-source code[1] and the results are reported in Table VII. It can be seen that all EfficientNets are significantly improved by introducing DFMs.

4) Visualization: We visualize the activation-based attention maps [67] based on the last residual group of the original ResNet-50 and DF-ResNet-50 for images from the ImageNet validation set, as shown in Fig. 6. We can see that the attention maps of DF-ResNet-50 cover the target object regions better than those of ResNet-50, demonstrating that DF-integrated networks can utilize information from target object regions to aggregate more discriminative features under the guidance of the intermediate decisions made by DFMs.

E. Applications

To further demonstrate the generalization ability of DFMs, we extend our approach to two related image recognition tasks, i.e., object detection and semantic segmentation.

1) Object Detection: We conduct object detection experiments on the PASCAL VOC dataset [70] and the Microsoft COCO dataset [71]. We adopt RetinaNet [72] based on

[1]https://github.com/rwightman/pytorch-image-models

TABLE VIII
OBJECT DETECTION AP (%) OF RETINANET ON THE VOC 2007 TEST SET AND THE MICROSOFT COCO VALIDATION SET

VOC		Microsoft COCO	
Backbone	AP	Backbone	AP
ResNet-50	76.88	ResNet-50	32.12
DF-ResNet-50	**77.79**	DF-ResNet-50	**33.54**

TABLE IX
SEMANTIC SEGMENTATION MIOU (%) OF PSPNET ON THE VOC 2012 VALIDATION SET AND THE ADE20K VALIDATION SET

VOC		ADE20K	
Backbone	mIoU	Backbone	mIoU
ResNet-50	77.06	ResNet-50	41.85
DF-ResNet-50	**78.18**	DF-ResNet-50	**43.14**

the open-source code[2] as our detection method and compare the performance obtained by using ImageNet-pretrained ResNet-50 and DF-ResNet-50 as backbones. For VOC, we train the models on the union of the VOC 2007 trainval set and the VOC 2012 trainval set and evaluate them on the VOC 2007 test set. For COCO, we train the models on the training set and evaluate them on the validation set. The results reported in Table VIII show that RetinaNet with DF-ResNet-50 outperforms RetinaNet with ResNet-50. Since we use the same detection method, the gains can only be attributed to the enhanced representation power brought by DFMs, demonstrating the generalization ability of the proposed DFM for object detection.

2) Semantic Segmentation: We conduct semantic segmentation experiments on the PASCAL VOC dataset [70] and the ADE20K dataset Zhou-2017-ADE20K from CVPR 2017.[3] Specifically, we adopt PSPNet [73] based on the open-source code[4] as our segmentation method and compare the performance obtained by using ImageNet-pretrained ResNet-50 and DF-ResNet-50 as backbones. For VOC, we train the models on the VOC 2012 training set in addition to the augmented data with the annotation of [74] and evaluate them on the VOC 2012 validation set following [73]. For ADE20K, we train the models on the training set and evaluate them on the validation set. The mean of the classwise intersection over union (mIoU) results reported in Table IX show that PSPNet with DF-ResNet-50 outperforms PSPNet with ResNet-50, validating the generalization ability of the proposed DFM for semantic segmentation.

VI. CONCLUSION

We have presented DFM, a novel plug-in unit that can generate the category-coherent decisions based on the current layer of CNNs to guide the subsequent layers in image classification via decision fusion. DFNet generated by integrating DFMs into existing classification networks can be trained in an end-to-end fashion and show consistent improvements with minimal additional computational costs. Extensive comparisons validate the effectiveness and superiority of our approach. In addition, we have validated that the proposed DFM has sufficient generalization ability to improve performance on various other related image recognition tasks, e.g., object detection and semantic segmentation. We hope that DFM will become an important component of various network architectures.

Limitations and Future Work: First, we validate the efficacy of DFM only on well-balanced datasets. In the future, we may extend our approach to handle the long-tailed image classification tasks. Second, the decision consistent loss estimated instance by instance in a training batch may not be accurate. Calculating a more accurate loss would be an interesting problem to explore. Third, the current load-shuffle-split strategy requires manually setting the number of categories in each batch. Determining the optimal number of categories remains to be an open problem.

[2]https://github.com/zgcr/simpleAICV-pytorch-ImageNet-COCO-training
[3]doi: 10.1109/CVPR.2017.544
[4]https://github.com/hszhao/semseg

REFERENCES

[1] R. M. Haralick, K. Shanmugam, and I. Dinstein, "Textural features for image classification," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-3, no. 6, pp. 610–621, Nov. 1973.
[2] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong, "Locality-constrained linear coding for image classification," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 3360–3367.
[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[4] W. Luo, J. Li, J. Yang, W. Xu, and J. Zhang, "Convolutional sparse autoencoders for image classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 7, pp. 3289–3294, Jul. 2018.
[5] P. Tang, X. Wang, B. Shi, X. Bai, W. Liu, and Z. Tu, "Deep FisherNet for image classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 7, pp. 2244–2250, Jul. 2019.
[6] Y. Luo, D. Tao, C. Xu, C. Xu, H. Liu, and Y. Wen, "Multiview vector-valued manifold regularization for multilabel image classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 5, pp. 709–722, May 2013.
[7] D. A. Forsyth and J. Ponce, *Computer Vision: A Modern Approach*. Upper Saddle River, NJ, USA: Prentice-Hall, 2002.
[8] J. Chen *et al.*, "WLD: A robust local image descriptor," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1705–1720, Sep. 2010.
[9] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.
[10] Y. Pang, Y. Yuan, and X. Li, "Gabor-based region covariance matrices for face recognition," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 7, pp. 989–993, Jul. 2008.
[11] Y. W. Pang, W. Li, Y. Yuan, and J. Pan, "Fully affine invariant SURF for image matching," *Neurocomputing*, vol. 85, no. 15, pp. 6–10, May 2012.
[12] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, p. 27, 2011.
[13] J. Lu and E. Zhang, "Gait recognition for human identification based on ICA and fuzzy SVM through multiple views fusion," *Pattern Recognit. Lett.*, vol. 28, no. 16, pp. 2401–2411, Dec. 2007.
[14] S. Maji, A. C. Berg, and J. Malik, "Classification using intersection kernel support vector machines is efficient," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–8.
[15] O. Russakovsky *et al.*, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.
[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. NIPS*, 2012, pp. 1097–1105.
[17] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.
[18] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. CVPR*, Jul. 2017, pp. 4700–4708.

[19] A. Bilal, A. Jourabloo, M. Ye, X. Liu, and L. Ren, "Do convolutional neural networks learn class hierarchy?" *IEEE Trans. Vis. Comput. Graphics*, vol. 24, no. 1, pp. 152–162, Jan. 2018.

[20] W. Yu, K. Yang, H. Yao, X. Sun, and P. Xu, "Exploiting the complementary strengths of multi-layer CNN features for image retrieval," *Neurocomputing*, vol. 237, pp. 235–241, May 2016.

[21] P. Kontschieder, M. Fiterau, A. Criminisi, and S. R. Bulo, "Deep neural decision forests," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1467–1475.

[22] M. Mirza and S. Osindero, "Conditional generative adversarial nets," 2014, *arXiv:1411.1784*.

[23] S. Baek, K. In Kim, and T.-K. Kim, "Deep convolutional decision jungle for image classification," 2017, *arXiv:1706.02003*.

[24] V. N. Murthy, V. Singh, T. Chen, R. Manmatha, and D. Comaniciu, "Deep decision network for multi-class image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2240–2248.

[25] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[26] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. CVPR*, 2015, pp. 1–9.

[27] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Proc. ECCV*, 2016, pp. 630–645.

[28] S. Zagoruyko and N. Komodakis, "Wide residual networks," in *Proc. Brit. Mach. Vis. Conf.*, 2016, pp. 1–15.

[29] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6848–6856.

[30] A. G. Howard *et al.*, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.

[31] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.

[32] M. Tan and Q. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. ICML*, 2019, pp. 6105–6114.

[33] Z. Li, Y. Sun, L. Zhang, and J. Tang, "CTNet: Context-based tandem network for semantic segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Dec. 2, 2021, doi: 10.1109/TPAMI.2021.3132068.

[34] X. Xiang, Y. Zhang, L. Jin, Z. Li, and J. Tang, "Sub-region localized hashing for fine-grained image retrieval," *IEEE Trans. Image Process.*, vol. 31, pp. 314–326, 2022.

[35] J. Jiang, J. Liu, J. Fu, X. Zhu, Z. Li, and H. Lu, "Global-guided selective context network for scene parsing," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 4, pp. 1752–1764, Apr. 2022.

[36] Z. Li, J. Tang, and T. Mei, "Deep collaborative embedding for social image understanding," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 9, pp. 2070–2083, Jul. 2019.

[37] K. Grauman, F. Sha, and S. J. Hwang, "Learning a tree of metrics with disjoint visual features," in *Proc. NIPS*, 2011, pp. 621–629.

[38] S. Saha, G. Varma, and C. V. Jawahar, "Class2Str: End to end latent hierarchy learning," in *Proc. 24th Int. Conf. Pattern Recognit. (ICPR)*, Aug. 2018, pp. 1000–1005.

[39] A.-M. Tousch, S. Herbin, and J.-Y. Audibert, "Semantic hierarchies for image annotation: A survey," *Pattern Recognit.*, vol. 45, no. 1, pp. 333–345, 2012.

[40] J. Deng *et al.*, "Large-scale object classification using label relation graphs," in *Proc. ECCV*, 2014, pp. 48–64.

[41] Z. Yan *et al.*, "HD-CNN: Hierarchical deep convolutional neural networks for large scale visual recognition," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 2740–2748.

[42] W. Goo, J. Kim, G. Kim, and S. J. Hwang, "Taxonomy-regularized semantic deep convolutional neural networks," in *Proc. ECCV*, 2016, pp. 86–101.

[43] J. R. Quinlan, "Induction of decision trees," *Mach. Learn.*, vol. 1, no. 1, pp. 81–106, 1986.

[44] Z.-G. Liu, Y. Liu, J. Dezert, and F. Cuzzolin, "Evidence combination based on credal belief redistribution for pattern classification," *IEEE Trans. Fuzzy Syst.*, vol. 28, no. 4, pp. 618–631, Apr. 2020.

[45] P. Smets, "The combination of evidence in the transferable belief model," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 5, pp. 447–458, May 1990.

[46] G. Shafer, *A Mathematical Theory of Evidence*, vol. 1. Jan. 1976.

[47] F. S. Khan, R. M. Anwer, O. Torgersson, and G. Falkman, "Data mining in oral medicine using decision trees," *World Acad. Sci., Eng. Technol., Int. J. Comput., Elect., Automat., Control Inf. Eng.*, vol. 2, no. 1, pp. 225–230, 2008.

[48] Z.-H. Zhou and J. Feng, "Deep forest: Towards an alternative to deep neural networks," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Aug. 2017, pp. 3553–3559.

[49] K. Ahmed, M. H. Baig, and L. Torresani, "Network of experts for large-scale image categorization," in *Proc. ECCV*, 2016, pp. 516–532.

[50] T. Chen, S. Lu, and J. Fan, "SS-HCNN: Semi-supervised hierarchical convolutional neural network for image classification," *IEEE Trans. Image Process.*, vol. 28, no. 5, pp. 2389–2398, May 2019.

[51] C. Xiong, X. Zhao, D. Tang, K. Jayashree, S. Yan, and T.-K. Kim, "Conditional convolutional neural network for modality-aware face recognition," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 3667–3675.

[52] R. Kumar Srivastava, K. Greff, and J. Schmidhuber, "Highway networks," 2015, *arXiv:1505.00387*.

[53] C. Lee, S. Xie, P. W. Gallagher, Z. Zhang, and Z. Tu, "Deeply-supervised nets," in *Proc. AISTATS*, 2015, pp. 562–570.

[54] Z. Cai and N. Vasconcelos, "Cascade R-CNN: Delving into high quality object detection," in *Proc. CVPR*, Jun. 2018, pp. 6154–6162.

[55] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. ICML*, 2015, pp. 448–456.

[56] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *ICCV*, Dec. 2015, pp. 1026–1034.

[57] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7132–7141.

[58] M. Lin, Q. Chen, and S. Yan, "Network in network," 2013, *arXiv:1312.4400*.

[59] P. Goyal *et al.*, "Accurate, large minibatch SGD: Training ImageNet in 1 hour," 2017, *arXiv:1706.02677*.

[60] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," 2016, *arXiv:1611.05431*.

[61] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Proc. AAAI*, 2017, pp. 4278–4284.

[62] A. Paszke *et al.*, "Automatic differentiation in PyTorch," in *Proc. NIPS Autodiff Workshop*, 2017.

[63] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. ICML*, 2010, pp. 807–814.

[64] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jun. 2014.

[65] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," M.S. thesis, Univ. Toronto, Toronto, ON, Canada, Version 1.0, 2009.

[66] L. N. Darlow, E. J. Crowley, A. Antoniou, and A. J. Storkey, "CINIC-10 is not ImageNet or CIFAR-10," 2018, *arXiv:1810.03505*.

[67] S. Zagoruyko and N. Komodakis, "Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer," in *Proc. ICLR*, 2017, pp. 1–13.

[68] T. Elsken, J. H. Metzen, and F. Hutter, "Neural architecture search: A survey," *J. Mach. Learn. Res.*, vol. 20, no. 55, pp. 1–21, 2019.

[69] A. Howard *et al.*, "Searching for MobileNetV3," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1314–1324.

[70] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and W. Zisserman, "The PASCAL visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, Sep. 2010.

[71] T.-Y. Lin *et al.*, "Microsoft COCO: Common objects in context," in *Proc. ECCV*, 2014, pp. 740–755.

[72] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2980–2988.

[73] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2881–2890.

[74] B. Hariharan, P. Arbelaez, L. Bourdev, S. Maji, and J. Malik, "Semantic contours from inverse detectors," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 991–998.

Keke Tang (Member, IEEE) received the B.Eng. degree from Jilin University, Changchun, China, in 2012, and the Ph.D. degree from the University of Science and Technology of China, Hefei, China, in 2017.

He was a Post-Doctoral Fellow with The University of Hong Kong, Hong Kong. In 2019, he joined Guangzhou University, Guangzhou, China, where he is currently an Associate Professor. His research interests include the areas of robotics, computer vision, computer graphics, and cyberspace security. https://tangbohu.github.io/

Yuexin Ma received the B.Eng. degree in computer science from Shandong University, China, in 2015, and the Ph.D. degree from The University of Hong Kong, Hong Kong, in 2019.

She is currently an Assistant Professor with ShanghaiTech University, Shanghai, China, where she is the Leader of the 4DV Laboratory. Her research interests include computer vision and deep learning, with an emphasis on 3-D perception and reconstruction.

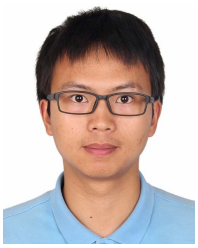
Dingruibo Miao received the M.S. degree from Guangzhou University, Guangzhou, China, in 2022. He is currently pursuing the Ph.D. degree with the State Key Laboratory of Remote Sensing Information Engineering for Surveying and Mapping, Wuhan University, Wuhan, China.

His research interests include computer vision and planetary science.

Peng Song received the Ph.D. degree from Nanyang Technological University, Singapore, in 2013.

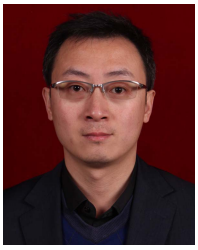
He was a Research Scientist with the Swiss Federal Institute of Technology Lausanne, Lausanne, Switzerland. In 2019, he joined the Singapore University of Technology and Design, Singapore, where he is currently an Assistant Professor of computer science. His research interest includes computer graphics, with an emphasis on geometry modeling, computational designs, and digital fabrication.

Dr. Song received SIGGRAPH Technical Papers Honorable Mention Award in 2022. He has served as a co-organizer of a virtual seminar series on computational fabrication in 2021 and 2022. He was invited to give a keynote talk at the Symposium on Solid and Physical Modeling 2022.



Zhaoquan Gu received the bachelor's and Ph.D. degrees in computer science from Tsinghua University, Beijing, China, in 2011 and 2015, respectively.

He is currently a Professor with the Department of Computer Science and Technology, Harbin Institute of Technology, Shenzhen, China, and a Researcher with the Department of New Networks, Peng Cheng Laboratory, Shenzhen. His research interests include wireless networks, distributed computing, big data analysis, and artificial intelligence security.



Zhihong Tian (Senior Member, IEEE) received the B.S., M.S., and Ph.D. degrees in computer science and technology from Harbin Institute of Technology, Harbin, China, in 2001, 2003, and 2006, respectively.

He served in different academic and administrative positions at the Harbin Institute of Technology. He is currently a Professor and the Dean with the Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou, Guangdong, China. He is also a part-time Professor with Carlton University, Ottawa, ON, Canada. He is honored as the Pearl River Scholar in Guangdong province. His research was supported by the National Natural Science Foundation of China, the National Key Research and Development Plan of China, and the National High-Tech Research and Development Program of China (863 Program). He has authored over 200 journal and conference papers. His research interests include computer networks and cyberspace security.

Dr. Tian is a Distinguished Member of the China Computer Federation. He also served as a member, the chair, and the general chair for a number of international conferences.

Wenping Wang (Fellow, IEEE) received the Ph.D. degree in computer science from the University of Alberta, Edmonton, AB, Canada, in 1992.

He was the Chair Professor and the Head of the Department of Computer Science, The University of Hong Kong, Hong Kong, from 2012 to 2017. He is currently a Professor and the Head of the Department of Visualization, Texas A&M University, College Station, TX, USA. His research interests include computer graphics, computer visualization, computer vision, robotics, medical image processing, and geometric computing.

Prof. Wang is an ACM Fellow. He received the John Gregory Memorial Award for his contributions in geometric modeling. He was the Chair of over 20 international conferences, including Pacific Graphics 2012, the ACM Symposium on Physical and Solid Modeling 2013, SIGGRAPH Asia 2013, and Geometry Submit 2019. He is an Associate Editor of several premium journals, including *Computer Aided Geometric Design*, *Computer Graphics Forum*, the IEEE TRANSACTIONS ON COMPUTERS, and the IEEE COMPUTER GRAPHICS AND APPLICATIONS.