Information Sciences 569 (2021) 496–507

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Modular neural network via exploring category hierarchy

Wei Han [a,b], Changgang Zheng [c], Rui Zhang [a,b], Jinxia Guo [a,b], Qinli Yang [d], Junming Shao [a,b,*]

[a] School of Computer Science and Engineering, University of Electronic Science and Technology of China, China
[b] Yangtze Delta Region Institute, University of Electronic Science and Technology of China, Huzhou, China
[c] Glasgow College, University of Electronic Science and Technology of China, China
[d] School of Resources and Environment, University of Electronic Science and Technology of China, China

ARTICLE INFO

Article history:
Received 3 February 2021
Received in revised form 14 April 2021
Accepted 11 May 2021
Available online 18 May 2021

MSC [2020]:
68T07
62H30

Keywords:
Modular neural network
Interpretable machine learning
Image classification
Category hierarchy
Learning to learn

ABSTRACT

Modular is a powerful and inherently hierarchical concept in the human brain to process a large variety of complex tasks. Converging evidence has shown several advantages to hierarchically modular network organizations in the human brain such as interpretability and evolvability of network function. Inspired by previous neuroscience studies, we propose MNN-CH, a novel modular neural network that is constructed with explored category hierarchy. The basic idea is learning to learn an optimized category hierarchy to decompose complex patterns. And specific patterns are imposed into corresponding modules to realize a transparent design of the neural network. Specifically, for a given classification task, each class or superclass is first represented as a prototype. Afterward, the category hierarchy is initially determined by investigating class similarity and gather similar ones to train each branch neural network (i.e., modular) separately. Finally, an error-driven prototype learning is introduced to refine the category hierarchy by updating the class-superclass affiliation. Experiment results on several image classification datasets show that our model has a good performance, especially in complex tasks. Beyond, we conduct an analysis to illustrate the tree-manner interpretability of the modular neural network.

1. Introduction

Modularity is a fundamental feature of many complex systems [1], like human brains. For example, there are certain groups of neurons in the visual cortex responding to specific functions [2]. And neurons within a minicolumn of cortical layers encode similar features [3]. Meanwhile, the segregation process of functional modules is discovered to promote network efficiency and executive functions [4], which is highly related to advanced cognitive control functions. Inspired by neuroscience studies, the modularity-based neural network (i.e., modular neural network) has been proposed and widely applied in fuzzy system [5,6], reinforcement learning [7,8], visual question answering [9,10], etc. The idea of the modular neural network is to split complex or overlapping patterns into sub-patterns, and each is learned by corresponding modules. In contrast to traditional neural networks, the modular neural network focuses on decomposed simple patterns within modules and conditionally activates different modules to handle various complex tasks. It thus is a potential way to be of many desirable properties such as effectiveness and interpretability.

* Corresponding author at: School of Computer Science and Engineering, University of Electronic Science and Technology of China, China.
E-mail address: junmshao@uestc.edu.cn (J. Shao).

https://doi.org/10.1016/j.ins.2021.05.032
0020-0255/

From the structural perspective, the modular neural network which employs modules to capture hierarchical patterns presents a tree-like information processing mechanism. It is a natural interpretable model in a tree manner, which is able to offer global and local interpretability by the function-transparent structure and conditionally activated path of modules, respectively. In terms of the functional aspect, each module focuses on specific simple sub-patterns, which results in a reasonable classification hyperplane in its own sub-tasks. The local distinguishability can be properly combined to provide solutions for global complex tasks. Therefore, the modular neural network is expected to be an effective model in the global task. In the construction of a modular neural network, the key point is to properly decompose patterns and impose them on each module.

To decompose task patterns, we adopt an intuitive instance-based constraint upon modules, in which a category hierarchy is constructed to model the sub-patterns. The basic hypothesis is that the representation of classes is similar since they have similar patterns. Therefore, it captures hierarchical relationships of patterns by gathering classes with similar patterns as a superclass. Previous works [11–13] employed some offline clustering methods to discover category hierarchy. As illustrated in Fig. 1, mammals, marine organisms, and household goods contain similar patterns. These patterns are captured by superclasses via collecting related classes and partially activated to jointly represent data during forwarding processing. However, how to exploit a proper category hierarchy is still an open question.

In the paper, we propose to explore an effective category hierarchy with an error-driven prototype learning method for modular neural network construction. In contrast to traditional offline clustering methods, the proposed method allows optimizing the category hierarchy iteratively during the training process. Specifically, we introduce the prototype-based representation for classes and superclasses, in which classes are first represented as prototypes. Then, we initialize the category hierarchy by investigating class similarity based on prototypes and gather similar ones to represent patterns. Building upon the error-driven prototype learning, the weights and the category hierarchy of the modular neural network are learned simultaneously. In the decision-making phase, for a given test instance, the modules are partially activated according to the correlation between the instance and patterns within modules to cooperate on the prediction. Therefore, once a sample is predicted by the modular neural network, the prediction is interpreted directly by tracing the functional modules responding in the processing. Experiments on image datasets show superior classification accuracy of ours model to the related methods, and the interpretability based on category hierarchy is demonstrated by interpretation analysis.

The rest of the paper is organized as follows. Section 2 describes interpretation methods of neural networks, modular neural networks as interpretable neural networks, and category hierarchy discovery for modular neural network construction. Section 3 introduces the architecture of the proposed model and category hierarchy fine-tuning. The experimental results and interpretation analysis are provided in Section 4, while Section 5 concludes the paper.

2. Related works

2.1. Interpretation methods of neural networks

Although the neural network model shows powerful performance in various areas, it suffers from doubts about black-box prediction. Therefore, interpreting neural networks is having increasing attention. The taxonomy of network interpretability consists of the post hoc and intrinsic ones. A post hoc interpretation applies methods to analyze the model after training. For instance, the high-layer filter was visualized by maximizing the activation [14], and the convolutional neural network layer was deconvoluted [15] for understanding. The network dissection method [16] labeled each neuron by comparing its receptive field with manual annotations and was extended to a generative adversarial network to manipulate semantic patterns in images [17]. A prediction for instances was interpreted by the activated location, which is obtained by weighted summation of activation maps in the last convolution layer [18] or the gradient back-propagation [19]. Meanwhile, the model-intrinsic interpretability is achieved by restricting the complexity of the machine learning model. The classical regularization methods are sparse coding [20] and low-rank constraints [21], decomposing and compressing overlapping representation. The priors regularize the search space of the model and improve the performance in applications [22–24]. Apart from the constraint on single-layer representation, patterns can also be decomposed and expressed in a part of the model as a functional module, i.e., the modular neural network in the study.

2.2. Modular neural network

According to where the constraint is imposed, the construction methods of the modular neural network are divided into instance-based, feature-based, and model-based methods. The model-based way only adopts a structurally interpretable model. For instance, the Mixture of Experts proposed by Hinton et al. [25] is the early work of modular neural networks. It adopted a gating network to automatically assign instances to modules and competitive learning to make modules independent. However, the search for pattern decomposition is unconstrained, and similar works include [26,27]. In contrast, the feature-based method offers a constraint on features to ensure pattern decomposition properly. For instance, Zhang et al. [28] proposed to regularize each filter in convolutional layers to respond to a local region. Meanwhile, a tree regularization [29] was proposed to encourages the complex decision boundaries to be well-approximated by human-simulatable functions. By reorganizing the last layer channels, the SEF algorithm [30] grouped semantic modules intuitively. Moreover,

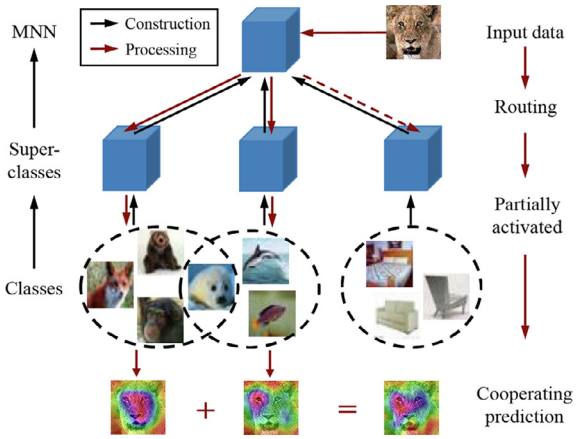

Fig. 1. The schematic diagram of category hierarchy construction and processing. Classes of the mammal, the marine organism, and the household are supposed to be of similar patterns and aggregated to superclasses, respectively. Since a class may include multiple patterns, such as marine mammals, the class sets of superclasses could be overlapping. The hierarchical relationship of classes and superclasses is called category hierarchy. When an image of a lion is input, superclasses with related patterns, such as the face structure and the eye, would be activated and cooperate to predict the current instance.

the hierarchical pattern decomposition has been applied in NLP and CV fields [31,24]. Complex patterns can be decomposed by hierarchical task segmentation, which refers to the instance-based approach. The instance-based method is a transparent design method to segment tasks and train modules with sub-tasks. For instance, in visual question answering [9,32] and reinforcement learning tasks [8,33], splitting tasks and building module network improve the model performance. Nevertheless, the task segmentation in such works needs expert knowledge. In classification tasks, another idea to automatically decompose a complex task is to cluster classes as superclasses, i.e. sub-tasks, and regularize modules with corresponding instances. The hierarchical relationship between classes and superclasses is called category hierarchy. In the study, we propose a novel effective category hierarchy discovery method.

2.3. Category hierarchy discovery

The category hierarchy captures hierarchical relationships of patterns by gathering classes with similar patterns as a superclass. In the exploration of category hierarchy, many interesting works have been put forward. The error-driven model [11] employed the spectral clustering as the basis of splitting branches and the validation error as the criterion to determine whether the current split is applicable. Meanwhile, the reconstruction error [34] from an individual auto-encoder of each task set was used as a relevance measure in continual learning. HD-CNN [12] adopted spectral clustering to discover superclasses, shared the low-level module of networks, and used overlapping superclass categories. Tree-CNN [13] introduced a branching strategy based on the confusion probability of new tasks upon previous superclasses. The relationship between classes could be captured by a confusion graph, and then, VT-CNN [35] built the visual tree by the community hierarchical detection algorithm. The hierarchical cluster validity index (i.e., HCVI [36]) was proposed to evaluate the cluster results for a proper category hierarchy. However, existing works consider the unsupervised category hierarchy discovery, which is no guarantee for current tasks. Beyond clustering approaches, our study aims to discover a category hierarchy with an error-driven prototype learning during the training.

3. Methodology

In this section, we demonstrate the modular neural network with category hierarchy exploring (i.e. MNN-CH). First, the overall architecture of the proposed model and its forward process are introduced. Then, we show how to organize the category hierarchy upon prototype-based representation. It decomposes complex patterns and guides the building of a modular neural network. Finally, the overall model is optimized end to end and the corresponding pseudo code is given.

3.1. Modular neural network architecture

A modular neural network is constructed with hierarchical functional modules to perform an end-to-end classification. Each module is an independent neural network and responds to a sub-task. An instance is predicted by corresponding branch modules routed by the root module. Without loss of generality, we build a two-layer modular network in this study. The structure (see Fig. 2) mainly consists of four parts: (*i*) the shared module M^S that encodes low-level features, (*ii*) a single independent component of the root module M_0^I that plays a role in routing, (*iii*) the category hierarchy described by the class-

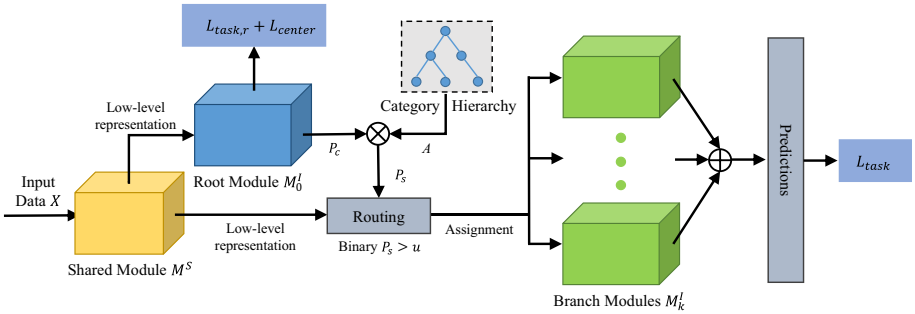

Fig. 2. The overview of the modular neural network via exploiting category hierarchy (MNN-CH). It consists of four parts, including the shared component M^S, a single independent component of root module M_r^l, the category hierarchy indicated by class-superclass affiliation A, and multiple independent components of branch neural networks $M_k^l, k \in \{1, \ldots, K\}$. The model performs an end-to-end classification with discovered category hierarchy. And we optimize model parameters and the category hierarchy alternatively with the gradients of losses.

superclass affiliation $A \in \mathbb{R}^{N \times K}$, (iv) multiple independent components of branch modules $M_k^l, k = \{1, \ldots, K\}$, that present a prediction of sub-tasks. Here, N and K are the number of classes and superclass, respectively. The shared module is reused for the root and branch modules. Since the front layers focus on class-agnostic low-level features, the sharing strategy is beneficial to reducing memory usage and computation consumption. Meanwhile, to simplify the model, the architecture of all independent components keeps the same.

In the feedforward, an instance x_i from the given dataset D is firstly fed into the shared module M^S to extract low-level features. The rest independent part of the root neural network M_0^l predicts the probability of each class P_c. Afterward, the superclass probability is computed by the product of class probabilities and the class-superclass affiliation matrix A, namely a mapping $[1, N] \rightarrow [1, K]$.

$$P_s = P_c \times A \tag{1}$$

Since the class set is overlapping, the sum of P_s for an instance may exceed 1, so an L_1 normalization is applied. Then, the low-level features from the shared module are diverted into the corresponding branch module(s) whose $p_{s,ik} > u$. The threshold value u is defined as a parametric variable $u = (\gamma K)^{-1}$, where γ is a parameter to be set. When the class sets of superclasses are disjoint, the algorithm degenerates into a divide-and-conquer method, and instances are only predicted by the branch that is of the maximum probability. In the overlapping setting, an instance x_i is given a prediction $\tilde{y}_i$ by averaging the prediction of the activated related branch module(s).

$$\tilde{y}_i = \frac{1}{|p_{s,i:} > u|} \sum_{k \in p_{s,i:} > u} M_k^l\left(M^S(x_i)\right) \tag{2}$$

3.2. Prototype-based representation for category hierarchy

The category hierarchy captures sub-task patterns by corresponding class sets, guiding the construction of the instance-based modular neural network. Intuitively, it builds a hierarchical relationship between classes and superclasses. A hypothesis is that the representation of classes is similar since they have similar patterns. Classes with similar patterns are aggregated as a superclass in the category hierarchy. Considering a class including multiple patterns, the class sets of superclasses should be overlapping. In previous works, the discovery of category hierarchy is unsupervised, and it is uncertain whether the pattern decomposed is proper. In the study, we propose to capture the class-superclass affiliation based on prototype-based representation and optimize it in the training, illustrated in Fig. 3. Specifically, the centers of classes and superclasses are represented as prototypes. Then, the category hierarchy is constructed with the similarity between the class center $C \in \mathbb{R}^{N \times d}$ and the superclass center $S \in \mathbb{R}^{K \times d}$. Here, d is the dimension of latent features. Namely, the class-superclass affiliation matrix $A \in \mathbb{R}^{N \times K}$ is determined by their similarity $Sim(C, S)$.

$$Sim(C, S) = h(-dist(C, S)) \tag{3}$$

$$A = \delta(Sim(C, S) > u) \tag{4}$$

where $dist(\cdot)$ is a distance metric and Euclidean distance is used in this study. The similarity is measured by the negative value of the metric and further normalized by a *softmax* function $h(\cdot)$. When the similarity is larger than the threshold value u, the element a_{jk} of the affiliation matrix is triggered to 1. Here, $\delta(condition) = 1$ if the *condition* is satisfied, $\delta(condition) = 0$ otherwise. The triggered element a_{jk} of the affiliation matrix indicates that the class j is in the class set of superclass k, namely $y_j \in Set_k$. The threshold value u is the mentioned parametric variable. Since the category hierarchy is defined on a

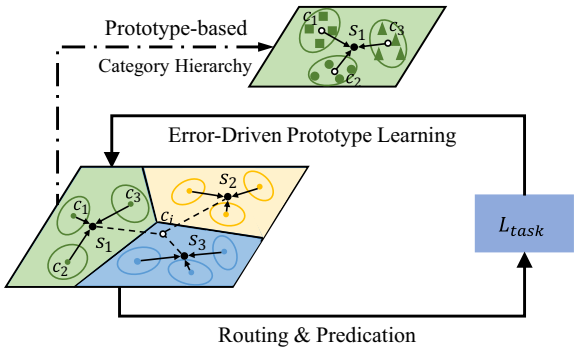

Fig. 3. The prototype-based category hierarchy and its updating. Intuitively, we take categories and superclasses as prototypes to describe their relationships. The center loss is introduced into the root module to constrain data structure and obtain class centers. Then, the superclass prototype is initialized by investigating the similarity of classes. The category hierarchy is defined on a continuous function with respect to the prototypes, therefore, it is differentiable and potential for learning to learn in the training.

continuous function with respect to the prototypes as a differentianle part, it is a promising way for learning to learn in the training.

Since we adopt the hard assignment to related modules, the category hierarchy could not update in training without the back-propagation of the task loss. In order to eliminate the non-differentiable part in the model, we conduct Backward Pass Differentiable Approximation (BPDA) [37] to handle the shattered gradient problem, namely, the hard routing. It applies some differentiable functions to approximate the non-differentiable ones in the backward pass for smooth and correct gradients. Specifically, the threshold function of superclass probability P_s is approximated as a sigmoid function $\sigma(\cdot)$ with a u shift in the backforwad. Meanwhile, the hard assignment is replaced by a masking operation in the calculation of back-propagation gradients.

$$\delta(p_{s,i} > u) \approx \sigma(p_{s,i} - u) \tag{5}$$

$$\begin{aligned} \tilde{y}_i &= \frac{1}{|P_{s,i:} > u|} \sum_{k \in P_{s,i:} > u} M_k^l\left(M^S(x_i)\right) \\ &= \frac{1}{|P_{s,i:} > u|} \sum_{k} M_k^l\left(M^S(x_i)\right) \odot \delta(p_{s,ik} > u) \end{aligned} \tag{6}$$

Here, $\odot$ is the dot product and the threshold $\delta(P_s > u)$ is acted as a masking matrix. The derivatives of these two approximate functions are easy to derive:

$$\frac{\partial \sigma(p_{s,i} - u)}{\partial p_{s,i}} = \frac{1}{e^{-(p_{s,i}-u)} + 1}\left(1 - \frac{1}{e^{-(p_{s,i}-u)} + 1}\right) \tag{7}$$

$$\frac{\partial \tilde{y}_i}{\partial \delta(p_{s,ik} > u)} = \frac{1}{|P_{s,i:} > u|} \sum_{k} M_k^l\left(M^S(x_i)\right) \tag{8}$$

3.3. Modular neural network training

The prototype representation of classes and superclasses is computed in the feature space of the root module. The root module routes samples to the related branch module(s) and should be trained first. The loss of the root module contains two parts. First, we employ the cross-entropy loss as the task loss of modules $L_{task,k}$, here $k \in \{r, 1, \ldots, K\}$.

$$L_{task,k} = -\sum_i y_i \log \tilde{y}_i \tag{9}$$

Apart from that, the center loss [38] is introduced to regularize the data structure of classes, which gathers data to their corresponding centers. In the center loss, the center matrix C consists of centers responding to each class and is intuitively regarded as the prototypes. The centers are represented in the feature space of the second to the last layer. Formally, given an instance x_i and label y_i, the root module predicts it as $\tilde{y}_i$. The objective of center loss L_{center} is to minimize the distance between the instance representation $\bar{x}_i$ and its corresponding center c_{y_i}. Finally, the root module parameters are optimized by the task loss and center loss with an imbalance coefficient λ.

$$L_{center} = \frac{1}{2}\sum_{i}^{N} \|\bar{x}_i - c_{y_i}\|_2^2 \tag{10}$$

$$L_r = L_{task,r} + \lambda L_{center} \tag{11}$$

Class centers are updated with gradients of center loss in step size α. The update equation of class center C could be derived as follows.

$$\frac{\partial L_{center}}{\partial x_i} = \bar{x}_i - c_{y_i} \tag{12}$$

$$\nabla c_j = \frac{\sum_{i=1}^{N} \delta(y_i = j) \cdot (c_j - \bar{x}_i)}{1 + \sum_{i=1}^{N} \delta(y_i = j)} \tag{13}$$

The center matrix C is randomly initialized and optimized in the training of the root module. Afterward, the superclass centers S are initialized by the K-Means algorithm upon the class centers.

Algorithm 1 MNN-CH

Input: Dataset D, supposed superclass number K, threshold parameter γ
Output: Optimized modular neural network
1: Train the root module and class prototype by Eq. (11)(13);
2: Initialize superclass prototypes;
3: Initialize category hierarchy with Eq. (4);
4: Train branch modules with Eq. (9);
5: **repeat**
6: Update model parameters with Eq. (14);
7: Update superclass prototypes S by gradients;
8: Update class prototypes C with Eq. (13);
9: Update the affiliation matrix A with Eq. (4);
10: **until** converge

Under the guidance of the initial category hierarchy, we use the trained root and branch modules to build the modular neural network. The next training of the whole model should consider multiple factors. The final prediction $\tilde{y}_i$ of the whole model is optimized by a cross-entropy loss as the task loss L_{task}. Meanwhile, the root module accuracy and the class prototypes should be simultaneously considered in the training. The category hierarchy is re-evaluated following the update of superclass prototypes, which could be performed with aggregated gradients of several batches or one. Finally, we optimize the whole model and category hierarchy guided with the overall loss $L_{overall}$, and the pseudo code of the whole process is summarized in Algorithm 1.

$$L_{overall} = L_{task} + L_{task,r} + \lambda L_{center} \tag{14}$$

4. Experiments

We evaluate MNN-CH with image classification on real-world datasets. Meanwhile, we attempt to analyze the discovered category hierarchy and investigate the decision behaviors of the modular neural network. All the experiments ran on an RTX 2080Ti card and were implemented by Pytorch 1.5.0 under Ubuntu 18.04.

4.1. Datasets

For image classification tasks, six datasets are adopted for assessment, including MNIST [39], CIFAR-10 [40], CIFAR-100 [40], Caltech-101 [41], Caltech-256 [42] and Tiny ImageNet.[1] MNIST is the early image classification dataset, consisting of 60,000 training grey images of handwritten digits with image size 28×28. CIFAR-10 and CIFAR-100 both include 32×32 color images and have 10 and 100 classes, respectively. These CIFAR datasets have 50,000 training images and 10,000 test images. Two Caltech datasets contain images sampled from the real world, which are of unfixed sizes and unbalanced class samples. Limited by computing resources, we scale the images to 64×64 to ensure a reasonable batch size. Caltech-101 consists of 101 classes and 8,677 images, while Caltech-256 consists of 256 classes and 30,609 images. Meanwhile, Tiny ImageNet is a sub-set of ImageNet [43] and is officially downsampled to 64×64. It involves images with 200 classes, and each class is pro-

[1] https://tiny-imagenet.herokuapp.com/

vided with 500 training images and 50 test images. For all datasets, 20% training data are divided as validation data. Data are augmented by random cropping and flipping, and normalized.

4.2. Baseline methods

Our method constructs an effective modular neural network via exploring category hierarchy and perform image classification based on module cooperation. We take a vanilla CNN as the baseline method and choose related state-of-the-art methods for comparison.

- **VGG16** [44]: The backbone of the vanilla convolutional neural network, i.e, VGG-16 with batch normalization layers. The classifier part is a full connection layer.
- **ED** [11]: This method grows branch modules with spectral clustering, and the determination of whether to apply the current structure is driven by the validation error.
- **Tree-CNN** [13]: The algorithm constructs category hierarchy upon the confusion probability of new classes with respect to existing super-classes.
- **HCVI** [36]: The paper proposes a hierarchical cluster validity index to improve the visual category tree learning. The network architecture adopts the shared low-level part.
- **VT-CNN** [35]: The approach introduces the confusion graph to capture the relationship between classes and then, builds the category hierarchy via the community hierarchical detection algorithm.
- **SEF** [30]: The method proposes to group semantic modules by reorganizing the last layer channels. The semantic groups are further enhanced to be activated on object parts with strong discriminability.
- **HD-CNN** [12]: An algorithm that adopts spectral clustering to discover category hierarchy, shares the low-level module, and uses overlapping super-class categories.
- **Ensemble**: An bagging ensemble of the same branch models as ours to show the effect of increasing model capacity on classification.

4.3. Parameter setting

The backbone of all methods employed the architecture of VGG-16. The shared parts of MNN-CH and HD-CNN occupied the first two convolutional blocks, while the other belonged to the independent part. The number of superclasses K was heuristically set as the square root of class numbers, and the parameter γ of threshold was tuned as 4 in all experiments. In the training of modules, the batch size was 128, and the weight λ of center loss was $5e-3$. Meanwhile, the optimizer SGD with an init learning rate of 0.1 was adopted. It was accompanied by a $0.2\times$ learning rate decay at {60, 120, 160} during 200 epochs. The step size α of center loss followed the original paper, i.e. 0.5. The optimizer for the category hierarchy adopted Adam, and the learning rate was set as $1e-3$. The update of the category hierarchy was suspended in the first 10 epochs for warm-up.

Hyper-parameters of baseline methods have been optimally tuned. Specifically, the parametric value γ of the category threshold in HD-CNN was 5. Meanwhile, the number K of HD-CNN kept the same as our method. Since the original work is based on the pre-trained network, all hyper-parameters of SEF were tuned to adapt training from scratch. The numbers of parts in the SEF algorithm were 2 for MNIST and CIFAR-10, 3 for CIFAR-100 and Caltech-101, 4 for Tiny ImageNet, and 5 for Caltech-256. Meanwhile, the balance weights of group loss, entropy loss, and knowledge distillation loss were set as 0.01, 0.05, and 0.01, respectively. For Error-Driven and Tree-CNN methods, they incrementally build category hierarchy, and we took 10-stage training to finish it. Therefore, the numbers of superclasses K for the two methods were adaptive. The split threshold of Tree-CNN was tuned as 0.55 for all datasets. Error-Driven split 2 leaves for each leaf node in each stage, as stated in the original work.

4.4. Results

The classification results are listed in Table 1. It could be observed that the proposed method MNN-CH achieves the best results in most datasets. It outperforms related methods and improves the classification accuracy of the ensemble networks by 1% to 4% in complex tasks. It worths noting that the benefits of our method are more pronounced with larger datasets, namely more complex tasks. It indicates that the strategy of MNN-CH is an effective way to handle complex tasks by proper decomposition. Meanwhile, the comparison methods, Error-Driven and Tree-CNN, are constructed with a disjoint category hierarchy, showing an inferior classification performance. This might be because, in such a case, although the branch module performs well, the experience risk is transferred to the root module. The situation also appears in the results of HCVI and VT-CNN algorithms, which leads to no gain in classification performance. Considering the distinguishability of semantic groups simultaneously, the method SEF achieves a performance advantage of about 1% in complex datasets. Furthermore, we show the advantages of the proposed model by visualizing the loss landscape [45] of models. In Fig. 4, MNN-CH presents a lower loss value in the distance and faster loss change around the optimal in complex tasks. This means that the proposed model has a better lower bound, and it is easier to converge to the optimal point.

Table 1
The classification accuracy of different algorithms on real-world datasets.

Methods	MNIST	CIFAR-10	CIFAR-100	Caltech-101	Tiny ImageNet	Caltech-256
N/ K	10/ 3	10/ 3	100/ 10	101/ 10	200/ 14	256/ 16
VGG16	99.40%	93.37%	72.01%	70.23%	54.83%	56.80%
ED	**99.46%**	92.78%	70.62%	71.92%	58.55%	56.85%
Tree-CNN	99.36%	92.86%	68.47%	70.94%	48.27%	50.43%
HCVI*	-	-	71.72%	-	-	-
VT-CNN*	-	89.51%	72.04%	-	-	-
SEF	99.42%	92.94%	72.13%	73.34%	55.93%	57.07%
Ensemble	99.44%	**93.75%**	76.27%	74.79%	63.02%	61.92%
HD-CNN	99.31%	93.02%	75.71%	76.56%	64.44%	65.19%
MNN-CH	99.38%	93.53%	**77.74%**	**78.97%**	**64.96%**	**66.31%**

* The results are the reported ones in original papars.

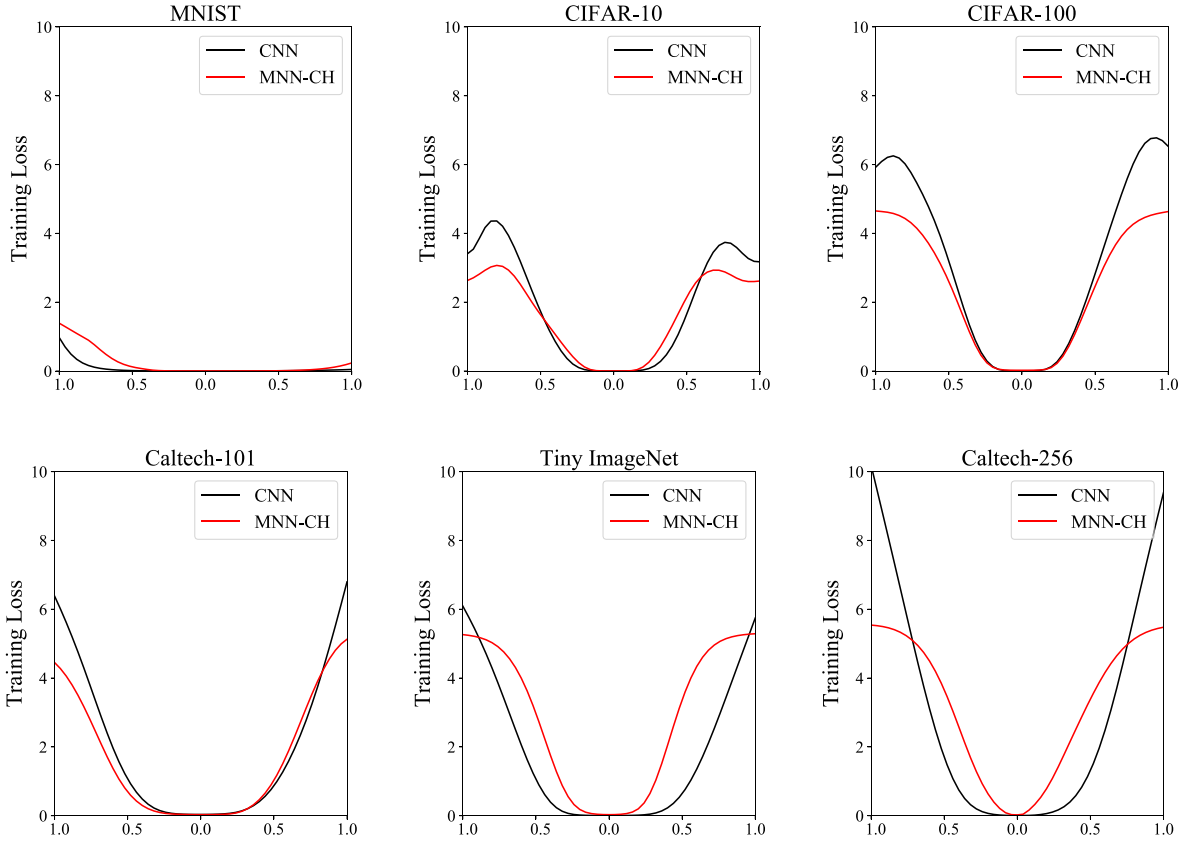

Fig. 4. The 1D loss landscape of the vanilla CNN and the proposed MNN-CH. The X axis indicates the random exploration of the parameter set, while the Y axis shows the training loss function value. The zero point represents the original parameter set. Therefore, the farther away from the zero point, the greater the change of the parameter set.

4.5. Ablation study

Since MNN-CH is a complex algorithm, we conduct an ablation study on CIFAR-100 to determine how much impact each component has on the results. Table 2 summarizes the results of MNN-CH with different components. In the table, although the accuracy of the vanilla convolutional neural network with center loss increased by 1%, MNN-CH without center loss seems to be unaffected. Specifically, if we adopt the same category hierarchy as the initial one in the MNN-CH and no center loss in the training, the performance gives only a slight decline. There is an accuracy gap from a single network to a multiple network ensemble, showing the great effect of model capacity. Beyond, the performance gain of about 0.8% compared to the ensemble method comes from the discovered category hierarchy with prototype-based representation. Meanwhile, the further optimization of category hierarchy brings about 0.7% boosting.

Table 2
An ablation study on CIFAR-100.

Method	Accuracy
Vanilla CNN	72.32%
+ center loss	73.18%
Ensemble	76.44%
MNN-CH	77.98%
− CH fine-tuning	77.27%
− center loss	77.19%

4.6. Sensitivity analysis

We perform a parameter sensitivity analysis of MNN-CH on CIFAR-100. There are three key parameters in our model: the number of superclasses K, the parameter of threshold γ, and the balance factor λ. To analyse their effect on MNN-CH, we varied K, γ and λ among $\{5, 10, 15, 20\}, \{1, 2, 3, 4, 5, 6\}$ and {1e-4, 5e-4, 1e-3, 5e-3, 1e-2, 5e-2, 1e-1}, respectively. We altered one of them while the others were fixed to the default value. The influence of K, γ, and λ on MNN-CH is presented in Fig. 5. The proposed method is robust to all γ, K, and λ. Although the classification accuracy is unsatisfactory with a small threshold parameter (i.e., 1), it is relatively stable after that point in a large range. The performance of MNN-CH increases following the number of superclasses, but the overall change is not significant. In order to balance the model performance and computation efficiency, we choose the square root of the class number as the number of superclasses, namely, 10 on CIFAR-100. Moreover, the classification accuracy slightly flutters with the change of balance factor λ.

4.7. Complexity analysis

Our approach introduces network-based function modules to build a tree structure. It leads to good interpretability and a linear increase of computational complexity upon module numbers. Specifically, we copied and extended the backbone network to several branch modules. Although the hard assignment to exact modules was adopted, CUDA does not offer supports to this situation. In the implementation, we used the masking operation to calculate the results. It means all paths of modules have to be traversed, and the hard assignment has no benefit on computational complexity. To reduce the computational complexity of the multi-level modular neural network, we took a part of the branch module as the shared part in modules of the same level. In the study, we built a two-level modular neural network with VGG16 as the backbone network. In the hierarchical segmentation, the first two of the five blocks in VGG16 were divided into shared modules. Namely, 30.85% Flops and 30.79% MAdd of the backbone network were reused in module computation. It makes the computational complexity of the proposed model increase linearly at a lower rate than directly K sub-networks. For instance, the ensemble of ten VGG16 networks in CIFAR-100 occupies 3.14GFlops and 6.28GMAdd, compared to 2.49GFlops and 4.97GMAdd of the proposed model.

4.8. Interpretation analysis

The decomposition strategy of the modular neural network upon category hierarchy leads to good interpretability in the tree manner. In this section, a simple analysis is performed about the discovered category hierarchy on CIFAR-100. The affiliation sets of superclasses are illustrated to investigate aggregated patterns from related classes. Meanwhile, we take some examples to demonstrate the explanation of the modular neural network and patterns which each module concerns.

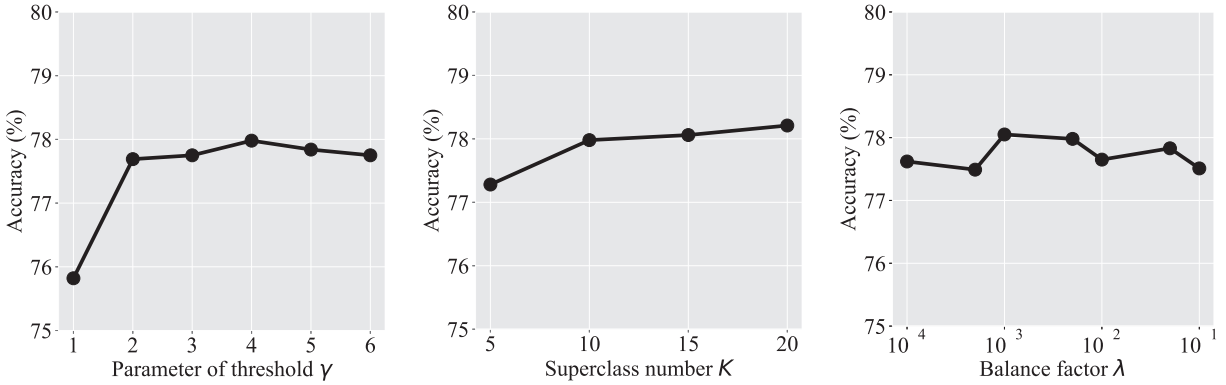

Fig. 5. The sensitivity analysis of model hyper-parameters on the CIFAR-100 data. The hyper-parameters includes the superclass number K and the threshold γ.

First, we assign each class only to the most related superclass, namely superclasses with the maximum probability, and summarize keywords to intuitively show their semantics, listed in Table 3. As shown, the semantic of each superclass seems to be clear. Specifically, the first superclass focuses on people and mammals. Although the second one could be activated by various classes, all these classes contain a blue background. The third superclass takes attention to mammals. Vehicles could be captured by the fourth superclass. The fifth superclass is of interest in sceneries. Insects and reptiles are the goals of the sixth. Household electrical devices and furniture could be processed by the seventh. Trees, fruits as well as vegetables, and colorful flowers belong to the last three superclasses, respectively. Based on the semantics of each branch module, the prediction of an instance can be understood by tracing the activated module(s).

To intuitively demonstrate these patterns in modules, we use Grad-CAM [19] to visualize the activation of modules, illustrated in Fig. 6. Images with the same class and pose are sampled, and we employ the classes with different complexity, i.e., different numbers of dependent superclasses. As shown, the activation regions of the vanilla convolutional neural network are scattered on diverse parts of images, making it meaningless. In contrast, the ones of modules tend to focus on specific parts. Specifically, the maple tree is jointly represented by the superclasses related to trees and flowers. The tree-related superclass pays attention to the tree structure, and the flower-related superclass is attractive to colorful canopies, as well as the tulip stamen. Moreover, sharks could be decoupled into three superclasses, which are inferred to represent blue background, sceneries, and insects & reptiles. The first two superclasses are reasonable and concentrate on the shark body and background, respectively. Meanwhile, activation areas of the superclass aggregated from insects and reptiles surround the

Table 3
The semantics of superclasses. The classes are assigned to the most related superclass, and keywords of each superclass are summarized to demonstrate their semantics.

ID	Semantics	Affiliated classes
1	People & mammals	baby, boy, girl, hamster, man, mouse, rabbit, woman
2	Blue background	cloud, crocodile, dolphin, flatfish, ray, rocket, sea, seal, shark, skyscraper, trout, turtle, whale
3	Mammals	bear, beaver, camel, cattle, chimpanzee, dinosaur, elephant, fox, kangaroo, leopard, lion, otter, porcupine, possum, raccoon, shrew, skunk, squirrel, tiger, wolf
4	Vehicles	bus, lawn mower, motorcycle, pickup truck, tank, tractor
5	Sceneries	bicycle, bridge, castle, forest, house, keyboard, mountain, mushroom, plain, road, streetcar, train
6	Insects & reptiles	bee, beetle, butterfly, caterpillar, cockroach, crab, lizard, lobster, snail, snake, spider, worm
7	Household	bed, bowl, can, chair, couch, cup, lamp, plate, table, television, wardrobe
8	Trees	maple tree, oak tree, palm tree, pine tree, willow tree
9	Fruits & Circles	apple, bottle, clock, orange, pear, sweet pepper, telephone
10	Flowers & Colors	aquarium fish, orchid, poppy, rose, sunflower, tulip

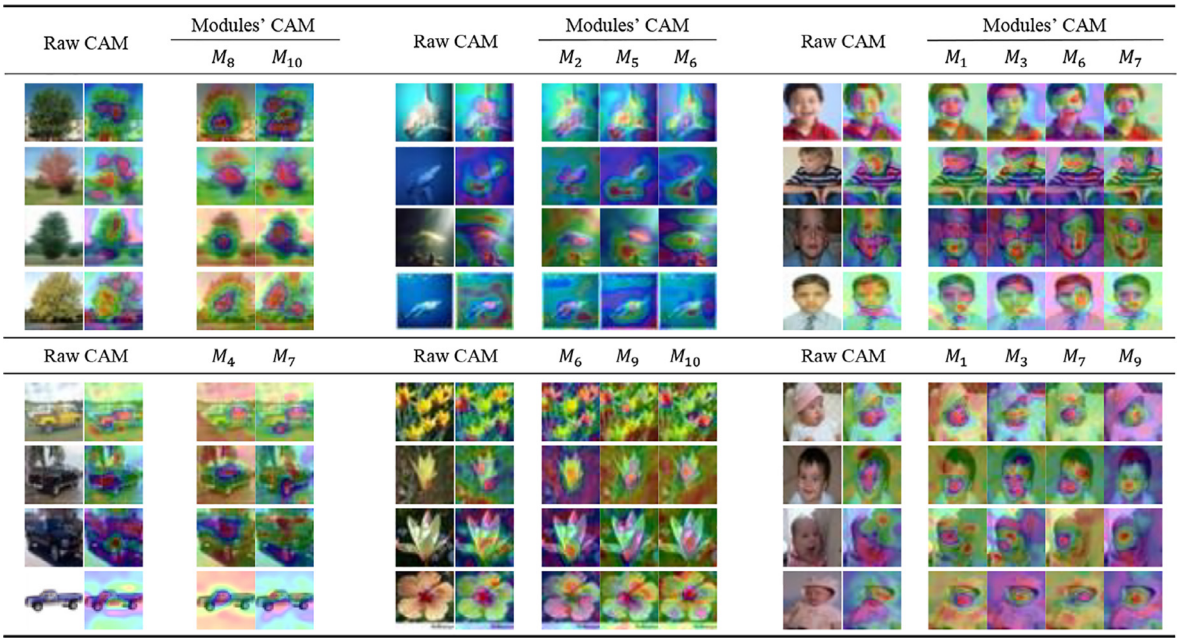

Fig. 6. The visualization of module activation via Grad-CAM. We employ images with the same class and classes in different complexity (i.e., different numbers of dependent superclasses) to investigate patterns of modules. The activation area is shown by the heatmap of HSV style on raw images.

shark body and boy's face. It seems hard to interpret with existing evidence, however, this module is also activated by the tulip. Insects and reptiles often attach themselves to subject surfaces such as tulip petals. Apart from that, the superclasses of people and mammals show attention to the face of boys and babies. Interestingly, the area near the eyes of boys, as well as baby's, and the wheels of vehicles precisely activate the module about households. It deserves further exploration, and one hypothesis is that the relevant pattern is the circular shape or light shadow. The vehicle superclass aims at the corresponding principle part. And the module assigned with fruit & circle samples shows a relationship with flowers in a scattered manner and the baby's face in reverse. The superclass based on specific categories pays attention to specific patterns in the decision-making. This shows that knowledge has been effectively decomposed by the modular neural network. Moreover, the decision-making of the modular neural network can be interpreted as a combination of module-related patterns, towards an interpretable model.

5. Conclusion

In this study, we propose a novel modular neural network construction method by exploring category hierarchy with error-driven prototype learning. Since the category hierarchy is automatically refined during the training process, MNN-CH is capable of decomposing a complex classification task into several sub-tasks properly. Simple sub-patterns lead to well-performing local modules in the sub-tasks which then cooperate for the global task. The modular neural network is beneficial from pattern decomposition and shows superior performance in image classification, especially in complex tasks. Moreover, inherited by the concept of modular, MNN-CH supports interpretation in a tree manner.

CRediT authorship contribution statement

Wei Han: Conceptualization, Methodology, Investigation, Software, Writing - original draft. **Changgang Zheng:** Validation, Writing - review & editing. **Rui Zhang:** Software. **Jinxia Guo:** Investigation. **Qinli Yang:** Writing - review & editing. **Junming Shao:** Conceptualization, Writing - review & editing, Supervision, Project administration.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work is supported by the Fundamental Research Funds for the Central Universities (ZYGX2019Z014), National Natural Science Foundation of China (61976044, 52079026), Fok Ying Tong Education Foundation (161062), and Sichuan Science and Technology Program (2020YFH0037).

References

[1] M.E. Newman, Modularity and community structure in networks, Proc. Natl. Acad. Sci. 103 (23) (2006) 8577–8582.
[2] S. Zeki, S. Shipp, The functional logic of cortical connections, Nature 335 (6188) (1988) 311.
[3] J.C. Horton, D.L. Adams, The cortical column: a structure without a function, Philos. Trans. R. Soc. B: Biol. Sci. 360 (1456) (2005) 837–862.
[4] G.L. Baum, R. Ciric, D.R. Roalf, R.F. Betzel, T.M. Moore, R.T. Shinohara, A.E. Kahn, S.N. Vandekar, P.E. Rupert, M. Quarmley, et al, Modular segregation of structural brain networks supports the development of executive function in youth, Curr. Biol. 27 (11) (2017) 1561–1572.
[5] P. Melin, D. Sánchez, O. Castillo, Genetic optimization of modular neural networks with fuzzy response integration for human recognition, Inf. Sci. 197 (2012) 1–19.
[6] F. Valdez, P. Melin, O. Castillo, Modular neural networks architecture optimization with a new nature inspired method using a fuzzy combination of particle swarm optimization and genetic algorithms, Inf. Sci. 270 (2014) 143–153.
[7] B. Fernandez-Gauna, M. Graña, J.M. Lopez-Guede, I. Etxeberria-Agiriano, I. Ansoategui, Reinforcement learning endowed with safe veto policies to learn the control of linked-multicomponent robotic systems, Inf. Sci. 317 (2015) 25–47.
[8] C. Devin, A. Gupta, T. Darrell, P. Abbeel, S. Levine, Learning modular neural network policies for multi-task and multi-robot transfer, in, in: 2017 IEEE International Conference on Robotics and Automation IEEE, 2017, pp. 2169–2176.
[9] J. Andreas, M. Rohrbach, T. Darrell, D. Klein, Neural module networks, in, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 39–48.
[10] Z. Yu, J. Yu, Y. Cui, D. Tao, Q. Tian, Deep modular co-attention networks for visual question answering, in, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 6281–6290.
[11] T. Xiao, J. Zhang, K. Yang, Y. Peng, Z. Zhang, Error-driven incremental learning in deep convolutional neural network for large-scale image classification, in, in: Proceedings of the 22nd ACM International Conference on Multimedia ACM, 2014, pp. 177–186.
[12] Z. Yan, H. Zhang, R. Piramuthu, V. Jagadeesh, D. DeCoste, W. Di, Y. Yu, Hd-cnn: hierarchical deep convolutional neural networks for large scale visual recognition, in, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 2740–2748.
[13] D. Roy, P. Panda, K. Roy, Tree-cnn: a hierarchical deep convolutional neural network for incremental learning, Neural Networks 121 (2020) 148–160.
[14] D. Erhan, Y. Bengio, A. Courville, P. Vincent, Visualizing higher-layer features of a deep network, University of Montreal 1341 (3) (2009) 1.
[15] M.D. Zeiler, R. Fergus, Visualizing and understanding convolutional networks, in: European conference on computer vision, Springer, 2014, pp. 818–833.
[16] D. Bau, B. Zhou, A. Khosla, A. Oliva, A. Torralba, Network dissection: Quantifying interpretability of deep visual representations, in, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 6541–6549.

[17] D. Bau, J.-Y. Zhu, H. Strobelt, B. Zhou, J.B. Tenenbaum, W.T. Freeman, A. Torralba, Gan dissection: Visualizing and understanding generative adversarial networks, in, in: Proceedings of the International Conference on Learning Representations (ICLR), 2019.
[18] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, A. Torralba, Learning deep features for discriminative localization, in, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2921–2929.
[19] R.R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, D. Batra, Grad-cam: Visual explanations from deep networks via gradient-based localization, in, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 618–626.
[20] J. Wright, A.Y. Yang, A. Ganesh, S.S. Sastry, Y. Ma, Robust face recognition via sparse representation, IEEE Trans. Pattern Anal. Mach. Intell. 31 (2) (2008) 210–227.
[21] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, Y. Ma, Robust recovery of subspace structures by low-rank representation, IEEE Trans. Pattern Anal. Mach. Intell. 35 (1) (2012) 171–184.
[22] J. Yu, Y. Rui, D. Tao, Click prediction for web image reranking using multimodal sparse coding, IEEE Trans. Image Process. 23 (5) (2014) 2019–2032.
[23] J. Yu, M. Tan, H. Zhang, D. Tao, Y. Rui, Hierarchical deep click feature prediction for fine-grained image recognition, IEEE Trans. Pattern Anal. Mach. Intell..
[24] C. Hong, J. Yu, J. Zhang, X. Jin, K.-H. Lee, Multimodal face-pose estimation with multitask manifold deep learning, IEEE Trans. Industr. Inf. 15 (7) (2018) 3952–3961.
[25] R.A. Jacobs, M.I. Jordan, S.J. Nowlan, G.E. Hinton, Adaptive mixtures of local experts, Neural Comput. 3 (1) (1991) 79–87.
[26] C. Fernando, D. Banarse, C. Blundell, Y. Zwols, D. Ha, A.A. Rusu, A. Pritzel, D. Wierstra, Pathnet: Evolution channels gradient descent in super neural networks, arXiv preprint arXiv:1701.08734..
[27] L. Kirsch, J. Kunze, D. Barber, Modular networks: Learning to decompose neural computation, in, Advances in Neural Information Processing Systems (2018) 2408–2418.
[28] Q. Zhang, Y. Nian Wu, S.-C. Zhu, Interpretable convolutional neural networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 8827–8836.
[29] M. Wu, M.C. Hughes, S. Parbhoo, M. Zazzi, V. Roth, F. Doshi-Velez, Beyond sparsity: tree regularization of deep models for interpretability, in: Association for the Advancement of Artificial Intelligence, 2018, pp. 1670–1678.
[30] W. Luo, H. Zhang, J. Li, X.-S. Wei, Learning semantically enhanced feature for fine-grained image classification, IEEE Signal Process. Lett. 27 (2020) 1545–1549.
[31] J. Yu, C. Zhu, J. Zhang, Q. Huang, D. Tao, Spatial pyramid-enhanced netvlad with weighted triplet loss for place recognition, IEEE Trans. Neural Networks Learn. Syst. 31 (2) (2019) 661–674.
[32] D. Mascharka, P. Tran, R. Soklaski, A. Majumdar, Transparency by design: closing the gap between performance and interpretability in visual reasoning, in, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 4942–4950.
[33] J. Andreas, D. Klein, S. Levine, Modular multitask reinforcement learning with policy sketches, in: Proceedings of the 34th International Conference on Machine Learning-Volume 70, JMLR. org, 2017, pp. 166–175..
[34] R. Aljundi, P. Chakravarty, T. Tuytelaars, Expert gate: Lifelong learning with a network of experts, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 3366–3375.
[35] Y. Zheng, Q. Chen, J. Fan, X. Gao, Hierarchical convolutional neural network via hierarchical cluster validity based visual tree learning, Neurocomputing 409 (2020) 408–419.
[36] Y. Liu, Y. Dou, R. Jin, P. Qiao, Visual tree convolutional neural network in image classification, in: 2018 24th International Conference on Pattern Recognition (ICPR) IEEE, 2018, pp. 758–763.
[37] A. Athalye, N. Carlini, D. Wagner, Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples, in: International Conference on Machine Learning, 2018, pp. 274–283..
[38] Y. Wen, K. Zhang, Z. Li, Y. Qiao, A discriminative feature learning approach for deep face recognition, in: European conference on computer vision, Springer, 2016, pp. 499–515..
[39] Y. LeCun, The mnist database of handwritten digits, http://yann.lecun.com/exdb/mnist/..
[40] A. Krizhevsky, G. Hinton, Learning multiple layers of features from tiny images (Tech. rep), Citeseer, 2009.
[41] L. Fei-Fei, R. Fergus, P. Perona, Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories, in: 2004 conference on computer vision and pattern recognition workshop, IEEE, 2004, pp. 178–178..
[42] G. Griffin, A. Holub, P. Perona, Caltech-256 object category dataset..
[43] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A.C. Berg, L. Fei-Fei, ImageNet Large Scale Visual Recognition Challenge, International Journal of Computer Vision (IJCV) 115 (3) (2015) 211–252, https://doi.org/10.1007/s11263-015-0816-y.
[44] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, CoRR (abs/1409.1556.).
[45] H. Li, Z. Xu, G. Taylor, C. Studer, T. Goldstein, Visualizing the loss landscape of neural nets, Neural Information Processing Systems (2018) 6391–6401.