Relative Contribution Mechanism: A Unified Paradigm for Disassembling Convolutional Neural Networks

Anonymous authors
Paper under double-blind review

Abstract

With the tremendous development of CNNs, obtaining an available CNN classifier is more challenging due to the massive number of parameters and deep structure. Recently, an emerging model disassembling and assembling task (MDA-Task) has been proposed to obtain new models easily from the perspective of model reusing. However, the existing methods are usually slow or inaccurate. In this paper, we put forward a contribution paradigm for MDA-Task, which unifies existing model disassembling and assembling methods into a universal formulation. We first propose a relative contribution mechanism that the prediction results of the CNN classifier are decided by the larger contribution value. Then, the analysis and two discoveries of contribution allocation and aggregation procedures are given around the above mechanism. Based on the two discoveries, we introduce a contribution attribution based CNN disassembling technique composed of single-layer contribution attribution and backward accumulation attribution, which can effectively find the category-aware components in each layer and associated components in adjacent layers, respectively. In addition, a contribution rescaling based CNN assembling technique is devised for assembling the above disassembled category-aware components from different CNN classifiers, which can achieve comparable accuracy performance with original CNN classifiers. Experiments on five benchmark datasets with three mainstream CNN classifiers verify the effectiveness of the proposed contribution paradigm and demonstrate that the contribution attribution based CNN disassembling and assembling technique can achieve significant accuracy increases and faster speed than the existing methods.

1 Introduction

In the past decade, the Convolutional neural network (CNN) has been one of the widely used classification network architectures, which has achieved remarkable breakthroughs in the image classification task (Feng et al., 2019; He, 2020) such as face recognition (Khalajzadeh et al., 2014; Saxena & Verbeek, 2016), medical image classification (Seetha & Raja, 2018; Yu et al., 2021), digital recognition (Bhagyashree et al., 2021; Jain et al., 2021), etc. The powerful discrimination ability of the CNN classifier benefits from its massive number of parameters and deep structure. However, It is usually time- and resource-consuming to train an available CNN classifier.

Therefore, some works, including knowledge distillation (Fang et al., 2022; Gou et al., 2020; Lei et al., 2021; Shen et al., 2021; Song et al., 2022), and transfer learning (Agarwal et al., 2021; Riccardo et al., 2018; Rajesh & Manikanthan, 2017; Siddhant et al., 2018; Simoes et al., 2018; Zhuang et al., 2020), are developed for reusing trained deep models, which is expected to save the cost of time, training samples, and GPU resources. However, all the above existing methods require additional training time or training samples. Recently, Hu et al. (2022) proposed a novel and interesting Model Disassembling and Assembling Task (MDA-Task), which aims to disassemble the deep models into independent modules and assemble those modules into a new deep model without additional training cost like playing LEGO toys.

For MDA-Task, the existing method (Hu et al., 2022) adopted the first-order derivative of prediction w.r.t. the feature maps to find the category-aware model components, based on the consumption that the large derivative value indicates important features. However, the derivative actually reveals the sensitivity of features towards the prediction and also has the noisiness and discontinuity problem (Lee et al., 2021). Those defects bring out two disadvantages: slow processing speed and incorrect category-aware component identification.

In this paper, we first propose a relative contribution mechanism for the CNN classifier, where the prediction results of the CNN classifier are decided by the larger contribution value, which is the combination of essential features and category-aware parameters (Section 3). Then, we analyze the contribution propagation in the CNN classifier (Section 3.1) and yields two discoveries on the contribution allocation and aggregation (Section 3.2). Based on the analysis and discoveries, we put forward a contribution paradigm, which unifies existing model assembling and disassembling techniques into a universal formulation (Section 3.3). The contribution paradigm and the above two discoveries are the foundation for disassembling and assembling the CNN classifier in this paper.

In the disassembling stage, we introduce a contribution attribution technique for finding the category-aware components. The contribution attribution technique is composed of two parts: single-layer contribution attribution and backward accumulation attribution. The former adopts the normalized contribution indicator to distinguish whether an unit is important for the prediction of a specific category. The latter accumulates the unimportant units between adjacent layers in the backward propagation direction.

With the disassembled components for specific categories from different models with the same network architecture, a contribution rescaling technique is devised for assembling those components into a new model. Due to the different relative contributions for output in different models, we rescale the contribution of components from different models to the same level, which ensures that different categories of the assembled model can equally predict final results.

For the whole disassembling and assembing process, there is no additional cost of training samples and fine-tuning, which is the outstanding advantage of the MDA-Task. What's more, extensive experiments on five datasets with three mainstream CNN classifiers demonstrate that the proposed model disassembling and assembling techniques can achieve significant accuracy increases and faster speed than existing methods.

Our contribution is therefore proposing the relative contribution mechanism and giving two discoveries on the contribution aggregation and allocation, which are the foundation for the following model disassembling and assembling. Moreover, we put forward a contribution paradigm unifying existing model disassembling and assembling techniques into a universal formulation. The contribution attribution based disassembling technique and contribution rescaling technique are introduced for MDA-Task. Finally, we conduct extensive experiments on a wide domain of datasets with mainstream classifiers to verify the superior accuracy and speed compared to existing methods.

2 Related Work

The related works are *feature attribution*, *model pruning*, and *subnetwork identification*. Existing *feature attribution* techniques can be divided into four types: activation-based technique, gradient-based technique, perturbation-based technique, and Layer-wise Relevance Propagation (LRP). Activation-based techniques (Desai & Ramaswamy, 2020; Naidu & Michael, 2020; Wang et al., 2020; Zhou et al., 2016) calculate a group of weight and then sum the activation map to visualize important features. Similar to (Hu et al., 2022), gradient-based techniques (Chattopadhay et al., 2018; Feng et al., 2022; Omeiza et al., 2019; Selvaraju et al., 2020; Shrikumar et al., 2017; 2016; Simonyan et al., 2013; Song et al., 2019; Sundararajan et al., 2017) adopt gradient-related terms to locate important features. Perturbation-based techniques (Lengerich et al., 2017; Zeiler & Fergus, 2014; Zhou & Troyanskaya, 2015; Zintgraf et al., 2017) discriminate important features through removing, masking, or altering them and running a forward pass on the new input, measuring the difference with the original output. LRP propagates the final prediction backward in the neural network with the designed local propagation rule, which is based on the conservation property, what a neuron has received must be redistributed to the lower layer in equal amounts (Montavon et al., 2019). Montavon et al. (2019) surveyed the above LRP rules (LRP-0 (Bach et al., 2015), LRP-ϵ (Bach et al., 2015), LRP-γ (Montavon et al., 2019), LRP-$\alpha\beta$ (Bach et al., 2015), flat (Lapuschkin et al., 2019), ω^2-rules (Montavon et al., 2016), $z^{\mathfrak{B}}$-rule (Montavon et al., 2016)) and analyzed the difference and connection.

Different from the above feature attribution methods, we propose the forward relative contribution mechanism and give the analysis and experimental verification. Besides, we put forward a novel contribution paradigm, which aggregates the normalized relative contribution to find category-aware model parameters correctly and is devised for the model disassembling task.

For *model pruning* (Lecun, 1990), the core idea is to discriminate the important and unimportant model parameters for the task and then prune those unimportant parameters, which aims to reduce the storage requirement and speed up the inference process. Existing model pruning techniques (Choudhary et al., 2020; Lei et al., 2018) contain weight pruning (Han et al., 2015), neuron pruning (Srinivas & Babu, 2015), filter pruning (Li et al., 2017), and layer pruning (Chen & Zhao, 2018). Different from model pruning which discriminates the task-aware redundant model components, the proposed contribution paradigm aims to find the category-aware features or parameters.

For *subnetwork identification*, existing techniques can be divided into three types: during-training, extra-training, and after-training. During-training techniques (Liang et al., 2020; Li et al., 2020) focus on guiding the deep neural network to generate class-related pathways by adding the extra modules during the training stage. Extra-training techniques (Wang et al., 2018; Hu et al., 2021; Frankle & Carbin, 2018) usually use a trainable module or training strategy with extra training steps to discover the subnetwork of a pretrained model. After-training techniques (Khakzar et al., 2021; Hu et al., 2022) mainly apply the attribution analysis method to discover the critical pathways of a pretrained model. The above methods can also be regarded as weight-based (Liang et al., 2020; Frankle & Carbin, 2018), activation-based (Li et al., 2020; Wang et al., 2018; Hu et al., 2021), and gradient-based (Khakzar et al., 2021; Hu et al., 2022), respectively.

Actually, the gradient just reveals the sensitivity of the feature map towards the model prediction and has the noisiness and discontinuity problem (Lee et al., 2021). Meanwhile, in our analysis (Section 3.3), weight-based and gradient-based methods are mainly related to the filter's parameters, and activation-based methods usually lack the analysis of subsequent layer's parameters influence, leading them to the incompleteness of contribution, while the contribution paradigm fully considered the parameters and activation of a network. In addition, the above subnetwork identification techniques (except (Hu et al., 2022)) are mainly used for applications like network interpretation and adversarial samples detection, while the proposed contribution paradigm focuses on disassembling a pretrained model to solve the model reuse problem, and it can also be used for network interpretation.

3 Analysis and Discovery of Relative Contribution Mechanism

Relative Contribution Mechanism. *The prediction of the CNN classifier is decided by the larger contribution value*[1]*, which is the combination of essential features*[2] *and category-aware parameters*[3]*. The contribution values in the preceding layer will aggregate into the contribution values in the latter layer through the forward propagation process.*

3.1 Analysis of Relative Contribution

For a CNN classifier, a Softmax function usually follows the last linear layer, which turns the last layer's outcomes, aka logits, into probabilities that sum to one. The higher the outcomes, the higher the corresponding probability. That means the relatively large outcome decides the final classification result in the last layer, which is the most intuitive manifestation of the relative contribution mechanism. In the other layers of CNN models, the relatively large outcome decides the magnitude of important features. Therefore, the relative contribution phenomenons apply to the whole forward propagation process of CNN models.

The following section will clarify the relative contribution allocation and aggregation in the whole forward propagation process. Taking the propagation in the layer l as an example (as shown in

[1]The score that reflects the feature's importance to the next layer.

[2]The features which are corresponding to the larger contribution value.

[3]The parameters which are related to the essential features.

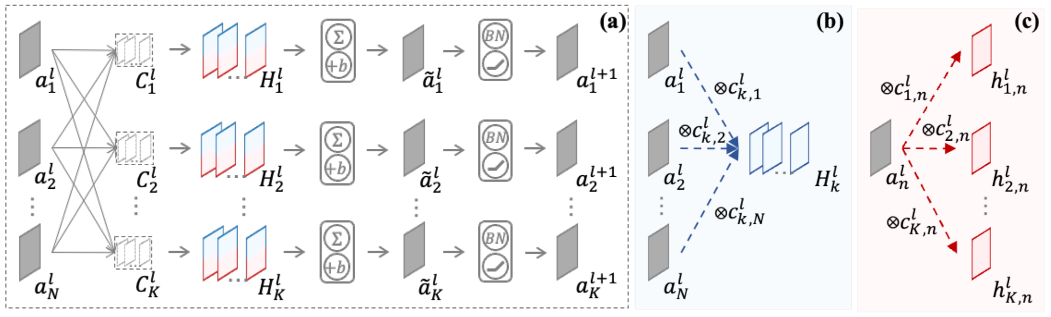

Figure 1: (a) the data flow diagram from the layer l to the layer $(l+1)$ for the convolutional network, where each convolution filter C_k^l is composed of a group of convolution kernels $C_k^l = \{c_{k,1}^l, c_{k,2}^l, ..., c_{k,n}^l, ..., c_{k,N}^l\}$. (b) the contribution aggregate to the k-th hidden feature maps H_k^l for the latent feature map $\widetilde{a}_k^l$. (c) the contribution allocation from the n-th activation map a_n^l.

Fig.1(a)), the input feature maps of the layer l are denoted as $\{a_1^l, a_2^l, ..., a_n^l, ..., a_N^l\}$, and the corresponding convolution filters are denoted as $\{C_1^l, C_2^l, ..., C_k^l, ..., C_K^l\}$, where each filter C_k^l is consists of N convolution kernels $C_k^l = \{c_{k,1}^l, c_{k,2}^l, ..., c_{k,n}^l, ..., c_{k,N}^l\}$.

With the convolution filter C_k^l, the hidden feature maps H_k^l of the layer l are calculated as follows:

$$H_k^l = \{h_{k,1}^l, h_{k,2}^l, ..., h_{k,n}^l, ..., h_{k,N}^l\}, h_{k,n}^l = c_{k,n}^l \otimes a_n^l, \tag{1}$$

where $\otimes$ denotes the partial convolution operation. With the hidden feature maps H_k^l, the output feature map $\widetilde{a}_k^l$ of the layer l is calculated as follows:

$$\widetilde{a}_k^l = \sum_{n=1}^{N} h_{k,n}^l + b_k, \tag{2}$$

where the b_k denotes the bias of the k-th channel. Then, the final activation map a_k^{l+1} of the layer $(l+1)$ is calculated as follows:

$$a_k^{l+1} = f(\widetilde{a}_k^l), \tag{3}$$

where f denotes the following operation, such as batch normalization, activation function, and pooling operation.

From Eqn.(2), we can see that the output feature map $\widetilde{a}_k^l$ of the layer l is summed by hidden feature maps H_k^l, which means that the value of $\widetilde{a}_k^l$ is determined by the value of single hidden feature map $h_{k,n}^l$. Like the softmax operation on the last layer, the larger the single hidden feature map, the more important to the output $\widetilde{a}_k^l$.

In the paper, the summed feature score $\uplus h_{k,n}^l$ of each hidden feature map $h_{k,n}^l$ is used to quantify the degree of contribution to the output $\widetilde{a}_k^l$, where $\uplus$ denotes the value summing operation of a matrix. For the classification task, positive/negative contributions have a positive/negative impact on the output. Therefore, the negative contribution score $\uplus h_{k,n}^l$ should be filtered out, which will reserve the positive contribution to the final classification results. In addition, the contribution to the outcome is relative as described before. So, the normalization operation is adopted to obtain the uniform contribution for different activation maps and model parameters. Finally, the normalized contribution score $r_{k,n}^l$ is given as follows:

$$r_{k,n}^l = \frac{v_{k,n}^l - min(V_k^l)}{max(V_k^l) - min(V_k^l) + \varepsilon}, V_k^l = \{v_{k,n}^l\}_{n=1}^N, v_{k,n}^l = \begin{cases} \uplus h_{k,n}^l & \uplus h_{k,n}^l > 0 \\ 0 & \uplus h_{k,n}^l \le 0 \end{cases}, \tag{4}$$

where ε denotes a minimum to avoid the situation of divide-by-zero.

For each output feature map $\widetilde{a}_k^l$, Fig.1 (b) shows the contribution aggregation process, where the contribution from activation maps $\{a_n^l\}_{n=1}^N$ are aggregated to hidden feature maps H_k^l with the k-th

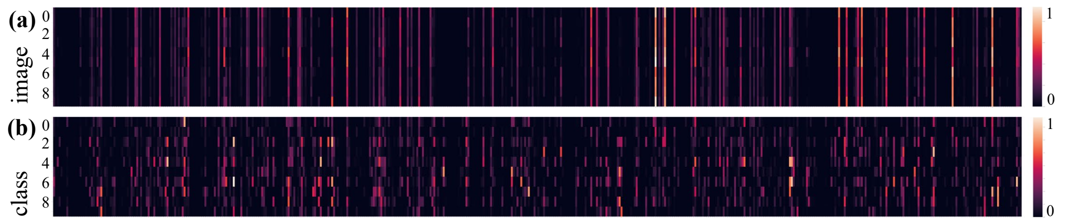

Figure 2: The pattern of contribution aggregation to an unit for samples of the same category (a) and different categories (b).

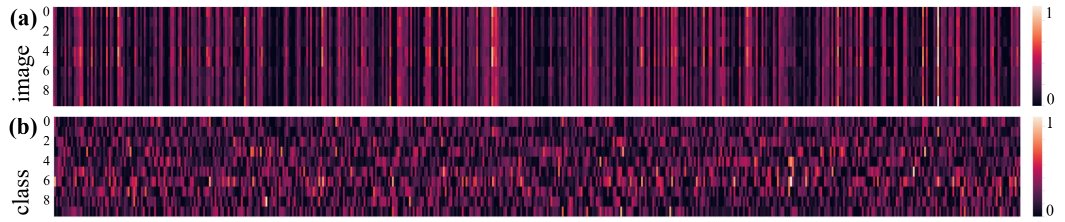

Figure 3: The total contribution pattern from different activation maps for samples of the same category (a) and different categories (b).

convolution filter C_k^l. The total contribution u_n^l from the activation map a_n^l is given as follows:

$$u_n^l = \sum_{k=1}^{K} r_{k,n}^l. \tag{5}$$

Fig.1 (c) shows the contribution allocation process, where the total contribution u_n^l from the activation map a_n^l are allocated to different hidden feature maps $\{h_{k,n}^l\}_{k=1}^K$ with different convolution kernels $\{c_{k,n}^l\}_{k=1}^K$.

3.2 Discovery of Relative Contribution

Discovery 1. *For the contribution aggregation to an unit, different samples of the same category have the similar relative contribution pattern, while different categories have different relative contribution pattern.*

Fig.2(a) shows the relative contribution pattern for different images of the same category bird in 13-th convolutional layer's 2-th hidden layer of VGG-16 (Simonyan et al., 2013) on CIFAR-10 (Krizhevsky, 2009). We can discover that different bird images have the almost same contribution pattern and some fixed contribution close to zero. Fig.2(b) shows the relative contribution of different categories in the same hidden layer, where we can see that the relative contribution pattern of categories varies with different categories.

Discovery 2. *For the total contribution of different units, different samples of the same category have the similar sparse total contribution pattern, which reveals that the prediction of a specific category highly rely on large contribution from fixed units (the combination of important feature maps and relevant model parameters). On the contrary, different categories have different total contribution pattern, which means that different categories rely on contribution from different units.*

Fig.3 shows the total contribution pattern from the layer 13's input feature maps of VGG-16 (Simonyan et al., 2013) on CIFAR-10 (Krizhevsky, 2009). From Fig.3(a), we can see that different images of the same category bird have similar total contribution pattern, which means that the prediction of category bird rely on the large contribution from the fixed input feature maps in the layer 13. For the convolution operation in the layer l, $a_k^{l+1} = \mathcal{F}(C_k^l \otimes \{a_1^l, a_2^l, ..., a_N^l\})$, where $\mathcal{F}$ denotes the following operation including summing, adding bias, batch normalization, activation function and e.t. after the convolution operation $\otimes$. Above equation indicates that with the same input activation maps $\{a_1^l, a_2^l, ..., a_N^l\}$ and $\mathcal{F}$, the output activation map a_k^{l+1} is only associated with the convolution filter C_k^l, which builds the relation between the k-th activation map with the the k-th convolution filter in the previous layer (Feng et al., 2021). Therefore, it reveals that a specific category is only related with fixed convolution filters combining the discovered relation between the specific category and the fixed activation maps in Fig.3(a), which is the foundation for the following CNN model disassembling. On the contrary, Fig.3(b) shows that different categories have different total contribution patterns, which indicates that prediction of different categories rely on large contribution from different activation maps. That is to way, different categories have their own relevant model parameters.

3.3 Discussion on Universality of Contribution Paradigm

As shown in Fig.1(a), the convolution operation in the layer l can be abbreviated to $a_k^{l+1} = \mathcal{F}(C_k^l \otimes \{a_1^l, a_2^l, ..., a_N^l\})$. For simply, according to the chain rule, the derivatives of outcome w.r.t. the input feature maps is calculated as follows:

$$\frac{\partial a_k^{l+1}}{\partial \{a_1^l, a_2^l, ..., a_n^l, ..., a_N^l\}} = C_k^l \mathcal{F}', \tag{6}$$

where $\mathcal{F}'$ denotes the derivatives of the operation after convolution. From Eqn.(6), we can see that the derivative of output w.r.t. the feature maps is mainly related to the convolution parameters C_k^l. Therefore, the first-order derivative attribution used in Hu et al. (2022) actually utilizes the approximation of large value parameters to find the category-aware model components.

From the perspective of relative contribution, the prediction of specific category is related with two terms: activation and model parameter, as given in Eqn.(1-5). Whereas, Hu et al. (2022) only adopt the large value model parameter (the approximation of gradient) to disassemble CNN classifier, which leads to incorrect identification of category-aware model components. In addition, some activation-based techniques (Desai & Ramaswamy, 2020; Naidu & Michael, 2020; Wang et al., 2020; Zhou et al., 2016) sum the activation map with a group of weight to visualize important features, which actually adopts approximation of activation to find important features. Most of those activation-based techniques can precisely discriminate category-aware important features, the root reason of which is neglect of the second term important model parameters. In summary, the contribution paradigm, which combines the essential features (the first term activation) and category-aware parameters (the second term). to find the prediction-related components, is an universal paradigm. It units the existing model disassembling and assembling methods.

4 Contribution Attribution based CNN Disassembling

Based on above relative contribution mechanism and analysis, we devise a contribution attribution technique to discriminate category-aware components. The contribution attribution technique is composed two steps: single-layer contribution attribution and backward accumulation attribution, which are devised to find the category-aware model components in each layer and accumulate the contribution influence between adjacent layers in the backward attribution process, respectively.

4.1 Single-layer Contribution Attribution

Based on the relative contribution mechanism, a threshold α is adopted to distinguish important and unimportant contribution $r_{k,n}^l$ calculated with Eqn.(4). Then, the importance indicator $q_{k,n}^l$ for the hidden feature map $h_{k,n}^l$ is calculated as follows:

$$q_{k,n}^l = \begin{cases} 1 & r_{k,n}^l \geq \alpha \\ 0 & r_{k,n}^l < \alpha \end{cases}, \tag{7}$$

where $q_{k,n}^l = 1$ denotes the single-layer hidden feature map $h_{k,n}^l$ is important, $q_{k,n}^l = 0$ denotes the single-layer hidden feature map $h_{k,n}^l$ is not important, and $\alpha \in (0, 1]$. Next, the discrete total contribution $\ddot{u}_n^l$ of the activation map a_n^l is summed as follows:

$$\ddot{u}_n^l = \sum_{k=1}^{K} q_{k,n}^l. \tag{8}$$

Similarly, based on the discovery on contribution allocated units, we adopt a threshold value β to filter out activation without contribution or with less contribution. Therefore, the contribution indicator t_n^l for the activation map a_n^l is calculated as follows:

$$t_n^l = \begin{cases} 1 & s_n^l \geq \beta \\ 0 & s_n^l < \beta \end{cases}, s_n^l = \frac{\ddot{u}_n^l}{max(\{\ddot{u}_1^l, \ddot{u}_2^l, ..., \ddot{u}_n^l, ..., \ddot{u}_N^l\}) + \varepsilon}, \tag{9}$$

where $t_n^l = 1/0$ denotes whether the activation map a_n^l has contribution for the next layer, and $\beta \in (0, 1]$, similarly, ε denotes a minimum to avoid the situation of divide-by-zero.

4.2 Backward Accumulation Attribution

In the forward propagation process, the small or negative aggregated contribution will be filtered out by the following activation function, which means that the small or negative aggregated contribution and the previous contribution allocated to the current unit do not contribute to the final prediction. In the disassembling process, importance indicators $\{q_{k,1}^l, q_{k,2}^l, ..., q_{k,n}^l, ..., q_{k,N}^l\}$ for the aggregated feature map H_k^l will be invalid if the contribution indicator t_k^{l+1} equals 0, which means the activation map a_k^{l+1} does not contribute to the final prediction. Therefore, the importance indicator $q_{k,n}^l$ (Eqn. (7)) for the hidden feature map $h_{k,n}^l$ with the accumulated influence from the activation map a_k^{l+1} is recalculated as follows:

$$q_{k,n}^l = \begin{cases} 1 & r_{k,n}^l \geq \alpha \text{ and } t_k^{l+1} > 0 \\ 0 & r_{k,n}^l < \alpha \text{ or } t_k^{l+1} = 0 \end{cases}. \tag{10}$$

Eqn.(8) and Eqn.(9) remain unchanged in the backward accumulation attribution. To sum up, the contribution indicator t_n^l (Eqn.(9)) for the activation map a_n^l in the layer l is affected by the contribution indicator t_n^{l+1} in the layer $(l + 1)$ by resetting $q_{k,n}^l$ to zero when $t_n^{l+1} = 0$.

The above two-step contribution attribution is also applicable for the fully connected layer. In addition, the backward accumulation attribution needs some tiny modification between the connection of the fully connected layer and the last convolution layer. More details about the contribution attribution in the fully connected layer are given in the *appendix*. In the last fully connected layer, the contribution to the final prediction of a specific category can be used to identify the important features. With the single-layer contribution attribution and the backward accumulation attribution, the contribution indicator t_n^l (Eqn.(9)) can be used to distinguish whether the activation map a_n^l has the contribution to the final prediction in the backward attribution process. As described in ***Discovery 2***, the activation map a_k^l is only associated with the convolution filter C_k^{l-1}. Therefore, the category-aware components can be effectively disassembled from the original CNN classifier with the above contribution attribution techniques.

5 Contribution Rescaling based CNN Asassembling

This paper only considers assembling category-aware components disassembled from homogeneous CNN classifiers trained on datasets with different categories. With the disassembled components from different models, the alignment padding strategy (Hu et al., 2022), which assembles category-aware components of different models through padding empty kernels for each convolution or linear filter, is adopted to assemble the model components in each layer. The padding operation ensures that all filters in each layer have the same number of kernels.

Furthermore, we devise a contribution rescaling strategy to optimize the component assembling in the last fully connected layer. Take the component assembling of two models as an example to elaborate on the contribution rescaling strategy. Assumed that there are two disassembled models m_1 with desired X categories (category number of the original classifier is large than X) and m_2 with desired Y categories, the disassembled category-aware model parameters of the last layer are denoted as $\{W_1^{m_1}, W_2^{m_1}, ..., W_x^{m_1}, ..., W_X^{m_1}\}$ and $\{W_1^{m_2}, W_2^{m_2}, ..., W_y^{m_2}, ..., W_Y^{m_2}\}$, and the output logits of the last layer are denoted as $\{e_1^{m_1}, e_2^{m_1}, ..., e_x^{m_1}, ..., e_X^{m_1}\}$ and $\{e_1^{m_2}, e_2^{m_2}, ..., e_y^{m_2}, ..., e_Y^{m_2}\}$. Then, with the contribution rescaling strategy, the new parameters $W'^{m_1}_x$ and $W'^{m_2}_x$ for the assem-

Table 1: The disassembling performance comparison. 'orig.' denotes the average accuracy of original models for specific categories ('Disassembled Category'). All scores are in %.

Dataset	Disassembled	VGG-16 (Simonyan et al., 2013)			ResNet50 (He et al., 2016)			GoogleNet (Szegedy et al., 2015)		
	Category	orig.	Ours	Grad.	orig.	Ours	Grad.	orig.	Ours	Grad.
MNIST	0	99.90	100.00	100.00	99.90	100.00	100.00	100.00	100.00	100.00
	1	100.00	100.00	100.00	99.91	100.00	100.00	100.00	100.00	100.00
	0-2	99.84	99.94	99.32	99.90	99.97	94.17	99.97	100.00	99.48
	3-9	99.57	99.67	99.12	99.54	99.59	98.73	99.69	99.77	99.72
FASHION-MNIST	0	89.30	100.00	100.00	90.40	100.00	100.00	91.10	100.00	100.00
	1	98.80	100.00	100.00	98.80	100.00	100.00	99.10	100.00	100.00
	0-2	93.53	97.90	94.67	93.73	98.00	97.80	94.07	97.57	93.43
	3-9	94.53	95.57	93.70	94.26	95.66	95.04	95.03	96.44	91.81
CIFAR-10	0	94.40	100.00	100.00	96.10	100.00	100.00	95.40	100.00	100.00
	1	96.50	100.00	100.00	96.20	100.00	100.00	97.40	100.00	100.00
	0-2	93.87	95.47	93.47	94.93	97.43	92.17	94.47	98.17	93.93
	3-9	92.49	92.27	88.59	93.81	94.46	94.36	93.46	93.17	91.93
CIFAR-100	0	84.00	100.00	100.00	92.00	100.00	100.00	90.00	100.00	100.00
	1	87.00	100.00	100.00	87.00	100.00	100.00	85.00	100.00	100.00
	0-19	71.05	82.50	76.80	75.55	77.15	75.45	75.90	87.55	81.40
	20-69	72.74	79.66	78.46	77.68	79.72	74.54	76.60	82.42	80.58
Tiny-ImageNet	0	82.00	100.0	100.00	92.00	100.00	100.00	88.00	100.00	100.00
	1	70.00	100.0	100.00	80.00	100.00	100.00	76.00	100.00	100.00
	0-69	50.17	55.49	50.89	56.06	56.40	52.83	52.89	59.40	59.31
	70-179	45.36	47.95	42.49	51.27	53.42	48.8	47.91	51.04	50.44

bled model in the last layer are calculated as follows:

$$W'^{m_1}_x = \frac{e}{e^{m_1}_x} W^{m_1}_x, W'^{m_2}_y = \frac{e}{e^{m_2}_y} W^{m_2}_y, e = \frac{\sum_{x=1}^{X} e^{m_1}_x + \sum_{y=1}^{Y} e^{m_2}_y}{X+Y}, \tag{11}$$

where e denotes the average logit value. The assembled model will predict the final results equally by combining the alignment padding strategy with the contribution rescaling strategy.

6 Experiments

In the experiments, we choose five benchmark datasets (MNIST (Lecun et al., 2001), Fashion-MNIST (Xiao et al., 2017), CIFAR-10 (Krizhevsky, 2009), CIFAR-100 (Krizhevsky, 2009), Tiny-ImageNet (Le & Yang, 2015)) and three mainstream CNN classifiers (VGG-16 (Simonyan et al., 2013), ResNet-50 (He et al., 2016), GoogleNet (Szegedy et al., 2015)) to verify the effectiveness of the proposed method for MDA-Task.

Parameter Setting. Unless stated otherwise, in convolutional layer, the default thresholds $\alpha = 0.3$, $\beta = 0.2$, in linear layer, the default thresholds $\alpha = 0.4$, $\beta = 0.3$, and the default disassembled layer number is all the layer. Besides, for the pretrained models, the default optimizer is SGD, the learning rate is 0.01, and the training epoch is 200. More details and are given in the *appendix*.

6.1 The Performance of Disassembling and Assembling CNN Classifier

The disassembling performance comparison results with 'Grad.' (Hu et al., 2022) are given in Table 1, where results are reported as an average of three runs to ensure fairness. From Table 1, we can see that the performance of single-class disassembling and multi-class disassembling by our method is close to or even above the original performance. What's more, our method achieves significant accuracy increases than 'Grad.', which verifies the effectiveness of the proposed contribution paradigm and the proposed contribution attribution techniques. Our method has smaller amount of model parameters (please refer to the *appendix* for the details quantitative results) than 'Grad.' as the significant accuracy increases, which indicates that the proposed attribution techniques can find category-aware model components more accurately. In addition, the proposed contribution attribution technique costs less time than 'Grad.' for the disassembling process. More experiment results about disassembling time-cost are given in *appendix*.

The assembling performance comparisons for categories from different datasets are given in Table 2, where we can see that the assembled new models by our method can achieve higher accuracy than 'Grad.', which demonstrates the significant advantage of the proposed method. More results and analysis are given in the *appendix*.

Table 2: The assembling performance for categories from different datasets. 'orig.' denotes the average accuracy for 'Assembled Category' from the original models. All scores are in %.

Dataset	Assembled	VGG-16 (Simonyan et al., 2013)			ResNet50 (He et al., 2016)			GoogleNet (Szegedy et al., 2015)		
	Category	orig.	Ours	Grad.	orig.	Ours	Grad.	orig.	Ours	Grad.
MNIST + FASHION-MNIST	0 + 0	94.60	97.35	50.00	95.15	62.05	50.00	95.55	96.05	50.00
	0-2 + 0-2	96.69	98.48	82.16	96.82	90.47	84.22	97.02	98.27	95.45
	0-2 + 3-9	96.12	95.62	81.15	95.95	88.58	84.22	96.51	96.34	86.12
	0-9 + 0-2	98.24	98.29	97.25	98.29	83.97	73.69	98.46	98.49	97.13
CIFAR-10 + CIFAR-100	0 + 0	89.20	50.00	50.00	94.05	77.30	50.00	92.70	99.60	94.25
	0-2 + 0-19	74.03	74.17	71.54	78.08	64.34	18.72	78.32	79.22	67.62
	3-9 + 20-69	75.16	73.72	69.97	79.66	72.03	54.62	78.67	70.24	69.25
	0-9 + 20-99	74.87	72.07	69.96	79.54	65.97	49.12	78.57	66.59	65.13
CIFAR-10 + Tiny-ImageNet	0 + 0	88.20	94.70	50.00	94.05	86.95	50.00	91.70	62.40	50.00
	0-2 + 0-69	51.97	53.20	50.75	57.65	43.09	5.98	54.59	57.51	45.20
	3-9 + 0-69	54.02	50.20	48.58	59.49	52.14	35.55	56.57	53.74	49.63
	0-9 + 70-179	49.33	42.30	26.19	54.85	47.21	28.88	51.73	48.00	44.02
CIFAR-100 + Tiny-ImageNet	0 + 0	83.00	50.00	50.00	92.00	53.00	50.00	89.00	50.00	50.00
	0-19 + 0-69	69.86	50.66	49.62	74.59	57.86	50.31	74.73	58.67	57.78
	20-69 + 70-179	71.48	50.08	33.94	76.54	56.53	49.68	75.08	54.09	53.46
	0-99 + 0-199	55.97	43.06	20.71	61.51	53.05	52.05	59.19	48.66	48.65

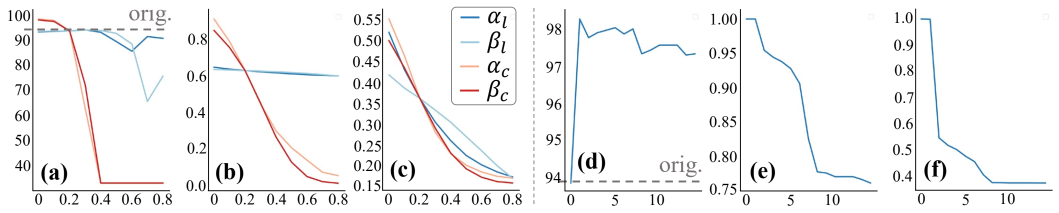

Figure 4: The ablation study on model hyper-parameters (α, β) and disassembled layers. The accuracy curve **(a)**, disassembling FLOPs ratio curve **(b)**, and model parameter amount ratio curve **(c)** with varying hyper-parameters in the linear (α_l, β_l) and convolutional (α_c, β_c) layers. The accuracy curve **(d)**, disassembling FLOPs ratio curve **(e)**, and model parameter amount ratio curve **(f)** with the accumulated disassembled layers in the backward direction.

6.2 Ablation Study

This section conducts the ablation study on the thresholds α and β in the linear layer and convolutional layer, as shown in Fig.4(a,b,c). In Fig.4(a), we can see that as the thresholds α and β increase, the accuracy of the disassembled model decreases, and thresholds in the convolutional layer is more sensitive than in the linear layer. Fig.4(b) shows that thresholds α and β have more significant changes on FLOPs in the convolutional layer than in the linear layer. In Fig.4(c), the model parameter amount ratio of the disassembled model will decrease with the thresholds increase.

Fig.4(d,e,f) show the ablation study on disassembled layer number, where the disassembling direction is from back to front. The accuracy will first rise and then fall as the layer number increases, as shown in Fig.4(d). From Fig.4(e,f), we can see that the model parameter amount and disassembling FLOPs ratio will decrease as the number of disassembled layers increases.

7 Conclusion

In this paper, we propose a relative contribution mechanism aka prediction results of a CNN classifier are decided by its larger contribution value, and give two discoveries on the pattern of contribution and allocation, respectively. Based on the above fundamental discoveries, we put forward a contribution paradigm, which unifies existing model disassembling and assembling techniques into a universal formulation. Next, we devise two attribution techniques to disassemble and assemble category-aware model components. Extensive experiments verify the effectiveness of the contribution paradigm and the superior accuracy and speed compared to the existing method. In the future, we will focus on disassembling CNN models for other tasks such as segmentation and detection, and assembling heterogeneous network architectures.

REFERENCES

N. Agarwal, A. Sondhi, K. Chopra, and G. Singh. Transfer learning: Survey and classification. *ICSICCS*, 2021.

S. Bach, A. Binder, G. Montavon, F. Klauschen, KR Müller, and W. Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS ONE*, 10, 2015.

P. M. Bhagyashree, L. K. Likhitha, and D. S. Rajesh. Handwritten digit recognition using deep learning. *IJSRST*, pp. 153–158, 2021.

Aditya Chattopadhay, Anirban Sarkar, Prantik Howlader, and Vineeth N Balasubramanian. Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In *WACV*, pp. 839–847, 2018.

S. Chen and Q. Zhao. Shallowing deep networks: Layer-wise pruning based on feature representations. *IEEE TPAMI*, pp. 1–1, 2018.

T. Choudhary, V. Mishra, A. Goswami, and J. Sarangapani. A comprehensive survey on model compression and acceleration. *Artificial Intelligence Review*, 53(3), 2020.

Saurabh Desai and Harish G. Ramaswamy. Ablation-cam: Visual explanations for deep convolutional network via gradient-free localization. In *WACV*, pp. 983–991, 2020.

Gongfan Fang, Kanya Mo, Xinchao Wang, Jie Song, Shitao Bei, Haofei Zhang, and Mingli Song. Up to 100 $\times$ faster data-free knowledge distillation. *AAAI*, 2022.

Z. Feng, W. Liang, D. Tao, L. Sun, and M. Song. Cu-net: Component unmixing network for textile fiber identification. *IJCV*, 127(10):1443–1454, 2019.

Z. Feng, J. Hu, S. Wu, X. Yu, J. Song, and M. Song. Model doctor: A simple gradient aggregation strategy for diagnosing and treating cnn classifiers. *AAAI*, 2022.

Zunlei Feng, Jiacong Hu, Sai Wu, Xiaotian Yu, Jie Song, and Mingli Song. Model doctor: A simple gradient aggregation strategy for diagnosing and treating cnn classifiers. *arXiv preprint arXiv:2112.04934*, 2021.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

J. Gou, B. Yu, SJ. Maybank, and D. Tao. Knowledge distillation: A survey. *ICCV*, 2020.

Song Han, Jeff Pool, John Tran, and William J. Dally. Learning both weights and connections for efficient neural networks. *NeurIPS*, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pp. 770–778, 2016.

Yang He, Ping Liu, Ziwei Wang, Zhilan Hu, and Yi Yang. Filter pruning via geometric median for deep convolutional neural networks acceleration. In *CVPR*, pp. 4340–4349, 2019.

Z. He. Deep learning in image classification: A survey report. In *ITCA*, 2020.

Jiacong Hu, Jing Gao, Zunlei Feng, Lechao Cheng, Jie Lei, Hujun Bao, and Mingli Song. Cnn lego: Disassembling and assembling convolutional neural network. *arXiv preprint arXiv:2203.13453*, 2022.

Jie Hu, Liujuan Cao, Tong Tong, Qixiang Ye, Shengchuan Zhang, Ke Li, Feiyue Huang, Ling Shao, and Rongrong Ji. Architecture disentanglement for deep neural networks. In *ICCV*, pp. 672–681, 2021.

M. Jain, G. Kaur, M. P. Quamar, and H. Gupta. *Handwritten Digit Recognition Using CNN*. 2021.

Ashkan Khakzar, Soroosh Baselizadeh, Saurabh Khanduja, Christian Rupprecht, Seong Tae Kim, and Nassir Navab. Neural response interpretation through the lens of critical pathways. In *CVPR*, pp. 13528–13538, 2021.

Hurieh Khalajzadeh, Mohammad Mansouri, and Mohammad Teshnehlab. Face recognition using convolutional neural network and simple logistic classifier. In *SCIA*, pp. 197–207. Springer, 2014.

Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.

S. Lapuschkin, S Wäldchen, A. Binder, G. Montavon, W. Samek, and KR Müller. Unmasking clever hans predictors and assessing what machines really learn. *Nature Communications*, 10(1), 2019.

Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015.

Y. Lecun. Optimal brain damage. *NeurIPS*, 2(279):598–605, 1990.

Yann Lecun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Intelligent Signal Processing*, pp. 306–351, 2001.

Jeong Ryong Lee, Sewon Kim, Inyong Park, Taejoon Eo, and Dosik Hwang. Relevance-cam: Your model already knows where to look. *CVPR*, 2021.

J. Lei, X. Gao, J. Song, X. L. Wang, and M. L. Song. Survey of deep neural network model compression. *Journal of Software*, 2018.

Jie Lei, Zhao Liu, Mingli Song, Juan Xu, Jianping Shen, and Ronghua Liang. Flexible knowledge distillation with an evolutional network population. *ICME*, 2021.

Benjamin J. Lengerich, Sandeep Konam, Eric P. Xing, Stephanie Rosenthal, and Manuela M. Veloso. Visual explanations for convolutional neural networks via input resampling. *arXiv preprint arXiv:1707.09641*, 2017.

Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. *ICLR*, 2017.

Yuchao Li, Rongrong Ji, Shaohui Lin, Baochang Zhang, Chenqian Yan, Yongjian Wu, Feiyue Huang, and Ling Shao. Interpretable neural network decoupling. In *ECCV*, pp. 653–669. Springer, 2020.

Haoyu Liang, Zhihao Ouyang, Yuyuan Zeng, Hang Su, Zihao He, Shu-Tao Xia, Jun Zhu, and Bo Zhang. Training interpretable convolutional neural networks by differentiating class-specific filters. In *ECCV*, pp. 622–638. Springer, 2020.

G. Montavon, S. Lapuschkin, A. Binder, W. Samek, and KR Müller. Explaining nonlinear classification decisions with deep taylor decomposition. *Pattern Recognition*, 65:211–222, 2016.

G. Montavon, A. Binder, S. Lapuschkin, W. Samek, and KR Müller. *Layer-Wise Relevance Propagation: An Overview*. Explainable AI: Interpreting, Explaining and Visualizing Deep Learning, 2019.

Rakshit Naidu and Joy Michael. Ss-cam: Smoothed score-cam for sharper visual feature localization. *arXiv preprint arXiv:2006.14255*, 2020.

Daniel Omeiza, Skyler Speakman, Celia Cintas, and Komminist Weldemariam. Smooth grad-cam++: An enhanced inference level visualization technique for deep convolutional neural network models. *arXiv preprint arXiv:1908.01224*, 2019.

M. Rajesh and Sv Manikanthan. Annoyed realm outlook taxonomy using twin transfer learning. *IJPAM*, 2017.

G. Riccardo, M. Anna, R. Salvatore, T. Franco, G. Fosca, and P. Dino. A survey of methods for explaining black box models. *ACM Computing Surveys*, 51(5):1–42, 2018.

S. Saxena and J. Verbeek. Heterogeneous face recognition with cnns. In *ECCV*, 2016.

J. Seetha and S. S. Raja. Brain tumor classification using convolutional neural networks. *Biomedical and Pharmacology Journal*, 11(3):1457–1461, 2018.

Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. *IJCV*, 128(2):336–359, 2020.

C. Shen, X. Wang, Y. Yin, J. Song, S. Luo, and M. Song. Progressive network grafting for few-shot knowledge distillation. *AAAI*, 2021.

Chengchao Shen, Xinchao Wang, Jie Song, Li Sun, and Mingli Song. Amalgamating knowledge towards comprehensive classification. *ArXiv*, abs/1811.02796, 2019.

Avanti Shrikumar, Peyton Greenside, Anna Shcherbina, and Anshul Kundaje. Not just a black box: Learning important features through propagating activation differences. *arXiv preprint arXiv:1605.01713*, 2016.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *ICML*, 2017.

A. Siddhant, A. Goyal, and A. Metallinou. Unsupervised transfer learning for spoken language understanding in intelligent agents. *AAAI*, 2018.

R. S. Simoes, V. G. Maltarollo, P. R. Oliveira, and K. M. Honorio. Transfer and multi-task learning in qsar modeling: Advances and challenges. *Frontiers in Pharmacology*, 2018.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *ICLR Workshop*, 2013.

Jie Song, Yixin Chen, Xinchao Wang, Chengchao Shen, and Mingli Song. Deep model transferability from attribution maps. In *NeurIPS*, volume 32, pp. 6179–6189, 2019.

Jie Song, Ying Chen, Jingwen Ye, and Mingli Song. Spot-adaptive knowledge distillation. *TIP*, 2022.

S. Srinivas and R. V. Babu. Data-free parameter pruning for deep neural networks. *Computer Science*, pp. 2830–2838, 2015.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *ICML*, 2017.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, pp. 1–9, 2015.

Haofan Wang, Zifan Wang, Mengnan Du, Fan Yang, Zijian Zhang, Sirui Ding, Piotr Mardziel, and Xia Hu. Score-cam: Score-weighted visual explanations for convolutional neural networks. In *CVPR Workshops*, pp. 24–25, 2020.

Yulong Wang, Hang Su, Bo Zhang, and Xiaolin Hu. Interpret neural networks by identifying critical data routing paths. In *CVPR*, pp. 8906–8914, 2018.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Xiaotian Yu, Zunlei Feng, Yuexuan Wang, Thomas Li, Xiuming Zhang, and Mingli Song. Tendentious noise-rectifying framework for pathological hcc grading. *BMVC*, 2021.

Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *ECCV*, 2014.

B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. *IEEE Computer Society*, 2016.

Jian Zhou and Olga G Troyanskaya. Predicting effects of noncoding variants with deep learning–based sequence model. *Nature Methods*, 12(10):931–934, 2015.

F. Zhuang, Z. Qi, K. Duan, D. Xi, and Q. He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 2020.

Luisa M Zintgraf, Taco S Cohen, Tameem Adel, and Max Welling. Visualizing deep neural network decisions: Prediction difference analysis. In *ICLR*, 2017.

In the ***appendix***, we provide the detailed experiment setting, elaboration, more experiment results of CNN disassembling and assembling, the ablation study on contribution rescaling strategy in CNN assembling, applications of our disassembling and assembling framework on other tasks, and discussion about limitations. In addition, source codes are given in the *'code.zip'* file.

A DETAILED EXPERIMENT SETTING

In this section, we give the detailed parameter settings for the linear layer and the convolutional layer of three mainstream CNN classifiers (VGG-16 (Simonyan et al., 2013), ResNet50 (He et al., 2016), GoogleNet (Szegedy et al., 2015)) on five benchmark datasets (MNIST (Lecun et al., 2001), Fashion-MNIST (Xiao et al., 2017), CIFAR-10 (Krizhevsky, 2009), CIFAR-100 (Krizhevsky, 2009), Tiny-ImageNet (Le & Yang, 2015)). Table 3 shows the detailed parameter settings for different networks on different datasets.

Table 3: The detailed parameters setting in assembling stage.

Dataset	VGG-16 (Simonyan et al., 2013)		ResNet50 (He et al., 2016)		GoogleNet (Szegedy et al., 2015)	
	Linear	Conv.	Linear	Conv.	Linear	Conv.
MNIST	$\alpha = 0.35, \beta = 0.55$	$\alpha = 0.30, \beta = 0.20$	$\alpha = 0.35, \beta = 0.55$	$\alpha = 0.40, \beta = 0.20$	$\alpha = 0.25, \beta = 0.55$	$\alpha = 0.40, \beta = 0.15$
FASHION-MNIST	$\alpha = 0.36, \beta = 0.55$	$\alpha = 0.40, \beta = 0.20$	$\alpha = 0.35, \beta = 0.55$	$\alpha = 0.40, \beta = 0.20$	$\alpha = 0.25, \beta = 0.55$	$\alpha = 0.40, \beta = 0.15$
CIFAR-10	$\alpha = 0.40, \beta = 0.55$	$\alpha = 0.30, \beta = 0.25$	$\alpha = 0.25, \beta = 0.45$	$\alpha = 0.35, \beta = 0.18$	$\alpha = 0.25, \beta = 0.55$	$\alpha = 0.40, \beta = 0.15$
CIFAR-100	$\alpha = 0.25, \beta = 0.45$	$\alpha = 0.35, \beta = 0.16$	$\alpha = 0.25, \beta = 0.55$	$\alpha = 0.40, \beta = 0.20$	$\alpha = 0.25, \beta = 0.45$	$\alpha = 0.35, \beta = 0.16$
Tiny-ImageNet	$\alpha = 0.25, \beta = 0.45$	$\alpha = 0.35, \beta = 0.16$	$\alpha = 0.25, \beta = 0.55$	$\alpha = 0.40, \beta = 0.20$	$\alpha = 0.25, \beta = 0.45$	$\alpha = 0.35, \beta = 0.16$

B CNN DISASSEMBLING

B.1 CONTRIBUTION ATTRIBUTION IN THE FULLY CONNECTED LAYER

In the Section 4.2 of the submitted paper, there is a special situation that if the layer l is a convolutional layer, and the layer $(l+1)$ is a linear layer, assume that the inputs of the layer $(l+1)$ are denoted as $\{p_1^{l+1}, p_2^{l+1}, ..., p_j^{l+1}, ..., p_J^{l+1}\}$, and the importance indicators of the layer $(l+1)$'s inputs are denoted as $\{t_1^{l+1}, t_2^{l+1}, ..., t_j^{l+1}, ..., t_J^{l+1}\}$. In CNN classifier, the inputs of the layer $(l+1)$ is usually calculated with the following operations: Global Average / Max Pooling, Flatten.

For the Global Average / Max Pooling, the Global Average / Max Pooling will not change the channel's number of the layer l's outputs $\{\widetilde{a}_1^l, \widetilde{a}_2^l, ..., \widetilde{a}_k^l, ..., \widetilde{a}_K^l\}$ and the layer $(l+1)$'s inputs $\{p_1^{l+1}, p_2^{l+1}, ..., p_j^{l+1}, ..., p_J^{l+1}\}$, aka $K = J$. Therefore, the importance indicator t_j^{l+1} of the input p_j^{l+1} is equal to the importance indicators of H_k^l, aka if $t_k^{l+1} = 0$, then the importance indicators $\{q_{k,1}^l, q_{k,2}^l, ..., q_{k,n}^l, ..., q_{k,N}^l\} = 0$.

For the Flatten operation, the part of inputs of the layer $(l+1)$ is calculated as follows:

$$\{p_{(k-1)*q+1}^{l+1}, p_{(k-1)*q+2}^{l+1}, ..., p_{k*q}^{l+1}\} = Flatten(\widetilde{a}_k^l), \tag{12}$$

where $\{p_{(k-1)*i+1}^{l+1}, p_{(k-1)*i+2}^{l+1}, ..., p_{k*i}^{l+1}\} \in \{p_1^{l+1}, p_2^{l+1}, ..., p_j^{l+1}, ..., p_J^{l+1}\}$, i denotes the number of inputs of the linear layer that flatten from one feature map in the preceding layer. Therefore, the importance indicator $\widetilde{t}_k^l$ of $\widetilde{a}_k^l$ is calculated as follows:

$$\widetilde{t}_k^l = \begin{cases} 1 & \sum\limits_{j=(k-1)*i+1}^{k*i} (t_j^{l+1}) \geq \beta \\ 0 & \sum\limits_{j=(k-1)*i+1}^{k*i} (t_j^{l+1}) < \beta \end{cases}, \tag{13}$$

where β is a threshold parameter, which filters out unimportant output feature map $\widetilde{a}_k^l$ of the layer l. If the importance indicator $\widetilde{t}_k^l = 0$, then the importance of H_k^l will also be regarded as zero. In summary, the backward accumulation attribution can be applied to the entire network.

Table 4: The disassembling performance comparison with other metrics. 'FLOPs' and 'Para.' denote the FLOPs and parameter amount (unit is M) of the disassembled model, respectively. For 'FLOPs' and 'Para.', 'score1 / score 2' denotes the performance of 'Grad.' / ours. The value of 'Time' denotes the ratio of time-cost of 'Grad.' to the proposed method's time-cost. The time-cost means the consuming time of the disassembling process.

Dataset	Disassembled	VGG-16 (Simonyan et al., 2013)			ResNet50 (He et al., 2016)			GoogleNet (Szegedy et al., 2015)		
	Category	FLOPs	Para.	Time	FLOPs	Para.	Time	FLOPs	Para.	Time
MNIST	0	7.24 / 20.31	4.24 / 4.59	1.11	91.38 / 105.06	9.94 / 13.77	1.69	32.32 / 46.40	3.42 / 4.39	3.79
	1	7.64 / 11.96	4.21 / 2.28	1.11	77.30 / 92.52	8.78 / 10.32	1.69	30.70 / 44.81	3.27 / 4.25	3.79
	0-2	19.87 / 26.65	17.13 / 9.62	1.11	109.82 / 122.13	17.71 / 18.39	1.69	41.02 / 49.04	4.92 / 5.30	3.79
	3-9	28.68 / 30.79	29.34 / 14.84	1.11	123.90 / 126.62	22.85 / 21.85	1.69	48.74 / 49.64	5.64 / 5.73	3.79
FASHION-MNIST	0	16.35 / 20.71	6.09 / 6.32	1.29	100.32 / 115.49	11.87 / 15.38	1.81	31.31 / 47.49	3.62 / 4.75	3.45
	1	15.60 / 13.66	5.50 / 3.17	1.29	98.93 / 94.90	11.25 / 12.18	1.81	29.90 / 45.91	3.35 / 4.53	3.45
	0-2	25.07 / 26.79	17.48 / 10.97	1.29	118.80 / 122.49	18.82 / 18.90	1.81	39.38 / 48.84	4.76 / 5.31	3.45
	3-9	27.28 / 30.30	26.73 / 15.27	1.29	128.89 / 127.69	22.79 / 21.78	1.81	44.70 / 49.12	5.67 / 5.74	3.45
CIFAR-10	0	7.50 / 8.35	4.28 / 2.13	1.31	80.94 / 56.33	8.44 / 8.11	1.79	34.15 / 37.72	3.51 / 3.95	3.43
	1	7.61 / 11.85	3.73 / 2.35	1.31	80.24 / 77.19	7.97 / 8.41	1.79	33.40 / 40.99	3.44 / 4.21	3.43
	0-2	25.54 / 18.69	18.39 / 5.04	1.31	113.69 / 106.51	16.81 / 15.55	1.79	41.35 / 45.50	4.72 / 5.23	3.43
	3-9	29.18 / 27.36	26.54 / 13.48	1.31	123.60 / 116.74	21.15 / 20.70	1.79	44.32 / 46.74	5.33 / 5.73	3.43
CIFAR-100	0	7.27 / 11.42	3.50 / 3.09	1.08	23.85 / 33.62	2.80 / 3.20	1.74	29.00 / 33.13	3.08 / 3.16	2.99
	1	6.84 / 12.73	3.41 / 3.11	1.08	24.28 / 55.76	2.89 / 6.18	1.74	29.83 / 35.43	3.01 / 3.22	2.99
	0-19	31.35 / 30.76	31.64 / 26.81	1.08	110.70 / 109.51	22.06 / 17.45	1.74	42.92 / 45.21	5.41 / 5.29	2.99
	20-69	32.55 / 31.97	32.41 / 30.01	1.08	116.15 / 120.01	22.99 / 21.56	1.74	45.78 / 47.08	5.57 / 5.57	2.99
Tiny-ImageNet	0	3.23 / 6.62	2.81 / 1.63	1.05	26.03 / 49.98	3.07 / 5.12	2.25	32.24 / 35.38	3.23 / 3.13	3.17
	1	3.11 / 4.66	2.75 / 1.58	1.05	24.81 / 38.59	2.93 / 4.41	2.25	34.40 / 34.21	3.23 / 3.13	3.17
	0-69	32.52 / 32.07	32.95 / 29.40	1.05	122.29 / 126.98	23.46 / 22.43	2.25	51.19 / 50.80	6.25 / 6.17	3.17
	70-179	32.06 / 32.61	33.17 / 30.72	1.05	124.22 / 128.33	23.61 / 23.23	2.25	51.74 / 51.13	6.32 / 6.23	3.17

B.2 Performance comparison with other metrics

To demonstrate that the disassembling performance of the proposed method is better than 'Grad.' (Hu et al., 2022), we give a detailed comparison with FLOPs and the parameter amount of the disassembled model and the time-cost of the disassembling process, as shown in Table 4.

Apart from the higher accuracy obtained by the proposed method, from Table 4, we can see that the FLOPs and the parameter amount of the proposed method are both less than 'Grad.' in most cases. The proposed method can achieve both higher accuracy and larger FLOPS values than 'Grad.', which indicates that the proposed method can find category-aware components more accurately. In addition, another advantage of the proposed method is the faster speed of the disassembling process than 'Grad.', where the derivation operation is very time-consuming.

C CNN Assembling

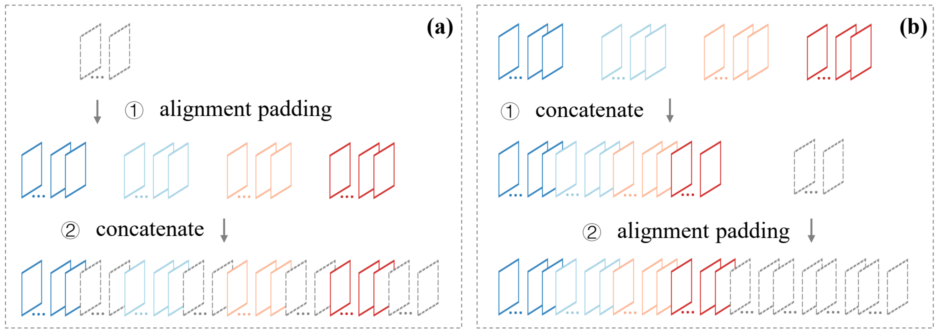

Figure 5: The (a) original alignment padding strategy and (b) improved alignment padding strategy in the inception module.

C.1 The assembling of GoogleNet

The alignment padding strategy (Hu et al., 2022) fails on GoogleNet (Szegedy et al., 2015) due to the particular multi-branch concatenation architecture aka inception module. As shown in Fig. 5(a),

Table 5: The ablation study on contribution rescaling strategy. 'Orig.' denotes the average accuracy for 'Assembled Category' from the original models. 'Asse.' and 'Ours' denote the accuracy of the disassembled classifier without and with contribution rescaling strategy, respectively. 'ep.+1' denotes the accuracy of the disassembled classifier with another one-epoch fine-tuning. All scores are in %.

Dataset	Assembled	VGG-16 (Simonyan et al., 2013)			
	Category	Orig.	Asse.	Ours	ep.+1
MNIST + FASHION-MNIST	0 + 0	94.60	53.32	97.35	100.00
	0-2 + 0-2	96.69	96.22	98.48	98.74
	0-2 + 3-9	96.12	89.51	95.62	96.63
	0-9 + 0-2	98.24	87.87	98.29	98.66
CIFAR-10 + CIFAR-100	0 + 0	89.20	50.00	94.00	98.80
	0-2 + 0-19	74.03	71.74	74.17	69.94
	3-9 + 20-69	75.16	71.97	73.72	54.13
	0-9 + 20-99	74.87	69.12	72.07	61.73
CIFAR-10 + Tiny-ImageNet	0 + 0	88.20	50.00	94.70	97.95
	0-2 + 0-69	51.97	52.41	53.20	54.88
	3-9 + 0-69	54.02	49.72	50.20	46.34
	0-9 + 70-179	49.33	41.54	42.30	35.31
CIFAR-100 + Tiny-ImageNet	0 + 0	83.00	50.00	85.50	60.50
	0-19 + 0-69	69.86	50.37	50.66	45.61
	20-69 + 70-179	71.48	50.06	50.08	40.33
	0-99 + 0-199	55.97	40.83	43.06	34.77

the alignment padding is used in each convolutional filter by default, making the blank feature calculated by zero kernels scattered all over the channel at various locations after the concatenate operation. For the Inception Module, we put the alignment padding after the concatenate operation in the improved alignment padding strategy, as shown in Fig. 5(b).

C.2 Ablation study on contribution rescaling strategy

The results of the ablation study on the contribution rescaling strategy are given in Table 5. The experiments are conducted on different datasets with VGG-16. From Table 5, we can see that the method with the contribution rescaling strategy ('Ours') achieves higher accuracy than the method without the contribution rescaling strategy ('Asse.') on all datasets. For some special cases ('0 + 0'), 'Ours' achieves about 45% accuracy increase. The above ablation study results verify the effectiveness of the contribution rescaling strategy.

D Application on Other Task

Given some pre-trained CNN classifiers, the proposed model disassembling and assembling technique can achieve category-customizable model reuse without any additional fine-tuning operation. What's more, the proposed model disassembling and assembling technique can also be extended to other tasks, including model decision route visualization, model compression, knowledge distillation, transfer learning, incremental learning, etc. Due to the limited time, we show the practical application of model decision route visualization, model compression, and knowledge distillation in the following section.

D.1 Decision Route Visualization

The decision route denotes the average data flow routes of each category, which is spare and fixed. As we described in the submitted paper, the prediction of different categories relies on significant contributions from different activation maps (aka different categories have their own relevant model parameters); the proposed contribution attribution method can help us build this decision route for each category. With the data flow routes of different categories, the researcher can easily locate the wrong data flow for misclassified samples by comparing it with the decision routes of the corresponding category, which partly solves the poor explainability of the deep models. What's more, the

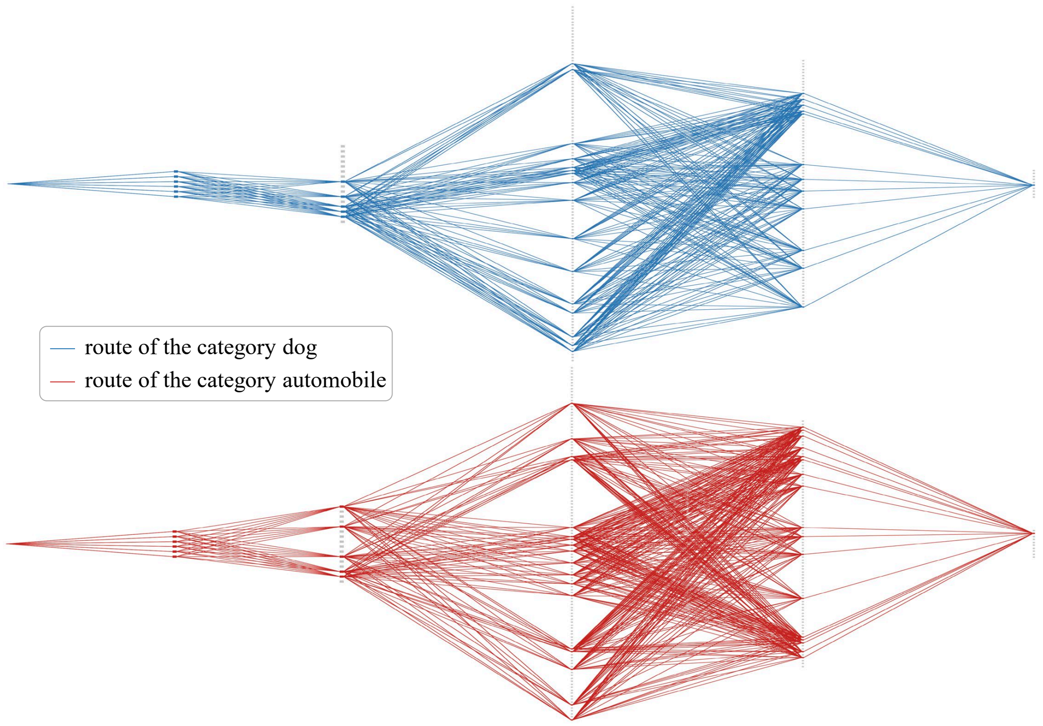

Figure 6: The decision route of category 'dog' and 'automobile' of LeNet-5 on CIFAR-10.

decision route visualization can be used as the master tool for better understanding and exploring the deep classifiers.

As shown in Fig. 6, we take the LeNet-5 (Lecun et al., 2001) (including two-layer convolution and three-layer linear) as an example to clearly demonstrate the decision route for specific categories, where we can see each category has its own decision routes. Different categories share some decision routes, which indicates that some model components are responsible for handling basic features for some categories.

In our model disassembling and assembling framework, the category-aware components (parameters) can be effectively disassembled from the original CNN classifier with the contribution attribution technique. It means that for a CNN classifier, we can disassemble the parameters that are used by all categories, and the parameters that are not used by any categories will be discarded. Therefore, our disassembling method can also be used for model compression.

D.2 Model Compression

In this section, we provide the performance of model compression by our method with VGG-16 (Simonyan et al., 2013) on two benchmark datasets (MNIST (Lecun et al., 2001), Fashion-MNIST (Xiao et al., 2017)). Besides, we also provide the performance comparison of model compression with the SOTA method FPGM (He et al., 2019), which implements the model compression by pruning. From Table 6, we can see that the compressed model by our method can achieve higher accuracy, and even higher than the method 'Grad.' (Hu et al., 2022) and FPGM (He et al., 2019) in most cases. However, it's obvious that the FLOPs ratio and parameter amount ratio of our method is higher than FPGM. The possible reason is that our contribution attribution method focuses on disassembling the relevant parameters of each category, while the pruning-based method focuses on filtering out irrelevant parameters for all categories. Therefore, our method reserves the commonly used parameters for all categories. However, the pruning-based method has the possibility of discarding some commonly used components, which does not influence the final prediction. Therefore, the pruning-based method (such as FPGM) has fewer FLOPs and parameter amounts. In future

Table 6: The model compression performance comparison.'Orig.' denotes the accuracy of the original model. 'Acc.' denotes the accuracy of the compressed model. 'rFLOPs' and 'rPara.' denote the floating point of operations ratio and parameter amount ratio of the disassembled model to the original model, respectively. All scores are in %.

Dataset	CNN		Ours			'Grad.' (Hu et al., 2022)			FPGM (He et al., 2019)		
	classifier	Orig.	Acc.	rFLOPs	rPara.	Acc.	rFLOPs	rPara.	Acc.	rFLOPs	rPara.
MNIST	VGG-16	99.65	99.68	95.28	62.32	99.33	90.99	94.98	99.45	87.79	90.59
	ResNet50	99.65	99.62	99.10	98.04	92.71	93.02	98.89	99.61	47.23	51.24
	GoogleNet	99.77	99.30	94.1	93.82	99.74	84.5	94.14	98.73	51.23	54.87
FASHION-MNIST	VGG-16	94.23	93.17	90.03	60.3	91.98	88.47	87.73	94.15	86.74	79.25
	ResNet50	94.10	93.92	98.75	96.13	93.98	99.43	99.11	93.87	46.19	52.31
	GoogleNet	94.74	94.59	93.86	94.45	94.32	90.9	93.5	94.52	50.74	53.67

Table 7: The knowledge distillation performance comparison. 'Orig.' denotes the average accuracy for 'Assembled Category' from the original models. (All accuracy scores are in %).

Dataset	Distilled	VGG-16 (Simonyan et al., 2013)			
	Category	Orig.	Ours	'Grad.' (Hu et al., 2022)	KA (Shen et al., 2019)
MNIST + FASHION-MNIST	0 + 0	94.60	97.35	50.00	92.24
	0-2 + 0-2	96.69	98.48	82.16	93.81
	0-2 + 3-9	96.12	95.62	81.15	92.76
	0-9 + 0-2	98.24	98.29	97.25	93.57
CIFAR-10 + CIFAR-100	0 + 0	89.20	94.00	50.00	84.35
	0-2 + 0-19	74.03	74.17	71.54	69.46
	3-9 + 20-69	75.16	73.72	69.97	71.38
	0-9 + 20-99	74.87	72.07	69.96	71.60
CIFAR-10 + Tiny-ImageNet	0 + 0	88.20	94.70	50.00	85.62
	0-2 + 0-69	51.97	53.20	50.75	52.67
	3-9 + 0-69	54.02	50.20	48.58	53.26
	0-9 + 70-179	49.33	42.30	26.19	51.84
CIFAR-100 + Tiny-ImageNet	0 + 0	83.00	85.50	50.00	81.31
	0-19 + 0-69	69.86	50.66	49.62	68.24
	20-69 + 70-179	71.48	50.08	33.94	67.52
	0-99 + 0-199	55.97	43.06	20.71	52.74

work, we will focus on investigating the potential ability of model disassembling and assembling on the model compression task.

D.3 Knowledge Distillation

In the model disassembling and assembling task, category-aware components disassembled from different models are assembled into a new model, which is similar to the knowledge amalgamating (KA) (Shen et al., 2019) that amalgamates knowledge from the multiple teachers into a single student by knowledge distillation. The goal of the above two tasks is similar. However, the adopted techniques are completely different.

In this section, we conduct comparison experiments of the proposed method and KA (Shen et al., 2019) with VGG-16 (Simonyan et al., 2013) on five benchmark datasets (MNIST (Lecun et al., 2001), Fashion-MNIST (Xiao et al., 2017), CIFAR-10 (Krizhevsky, 2009), CIFAR-100 (Krizhevsky, 2009), Tiny-ImageNet (Le & Yang, 2015)). From Table 7, we can see our method achieves higher accuracy than KA (Shen et al., 2019) in most cases, particularly in the case of less number of assembled categories, such as 'MNIST + FASHION-MNIST'. On the contrary, when the number of assembled categories increases (such as 'CIFAR-100 + Tiny-ImageNet'), the performance of the proposed method gets worse and is even lower than the accuracy of the original model and KA (Shen et al., 2019). The possible reason is that when the number of assembled categories increases, the category-aware model components for categories in 'dataset A' will increase. For an unknown test image, it will flow through all decision routes of the assembled model. Therefore, those components will disturb the correct prediction of samples in 'dataset B', which leads to decreased accuracy.

E Visualization of Contribution Attribution

In this section, Fig. 7-10 show more contribution attribution visualization results. From Fig. 7, we can see that in different layers of VGG-16, the different categories have different total contribution patterns. For a specific layer of different networks on different datasets, different categories

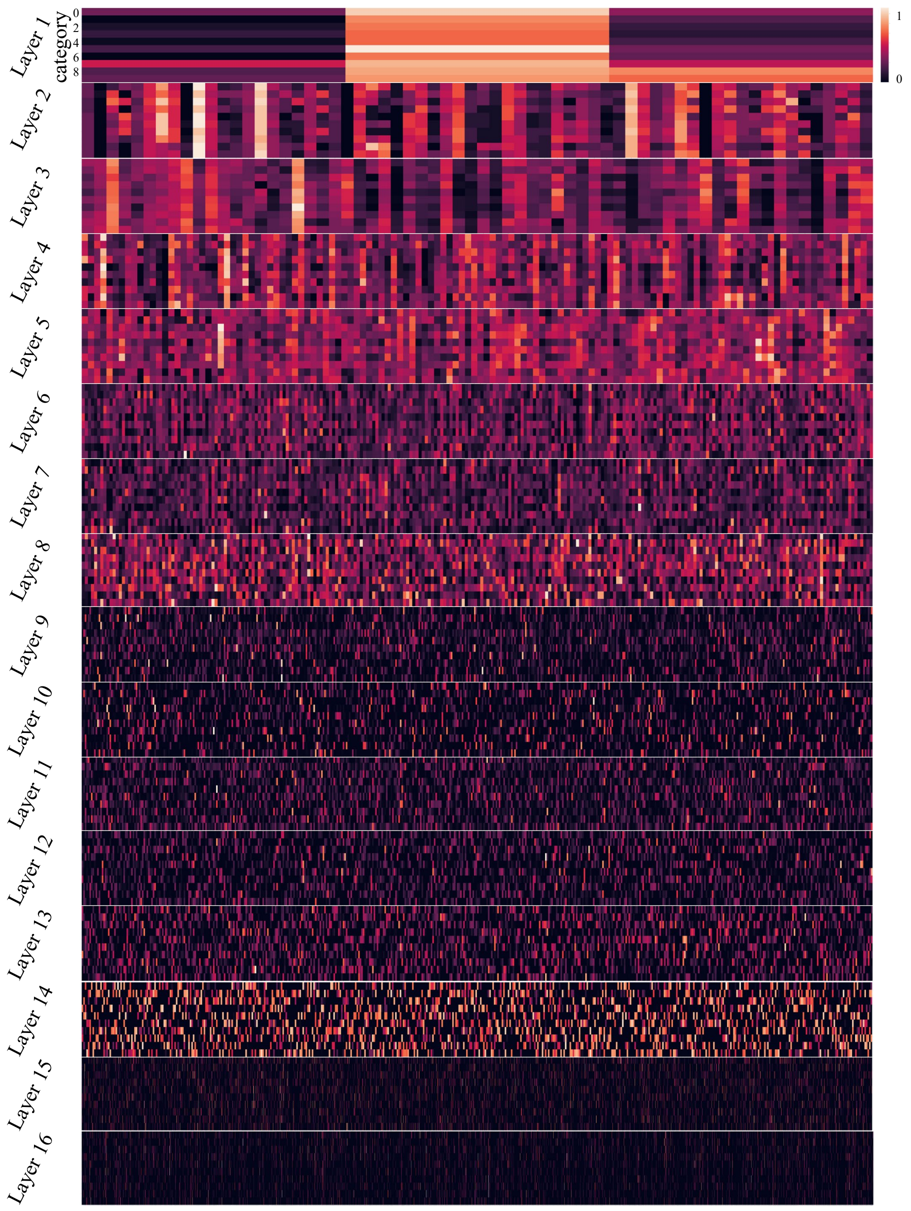

Figure 7: The total contribution patterns from different activation maps for samples of the different categories in different layers of VGG-16 on CIFAR-10.

also have different total contribution patterns, as shown in Fig. 8-10. In particular, for CIFAR-10, different categories have similar high responses on some activation maps in layer 66 of GoogleNet. However, there are still some activation maps with different responses for different categories, which is the determinant for the correct prediction of GoogleNet on CIFAR-10 dataset. Overall, those contribution attribution visualization results verify the second discovery on contribution allocated units again.

F Limitation and Future Work

Experiments in Table 1 of the submitted paper have shown that the model disassembled with the proposed method can achieve higher accuracy than the original model. For the assembling performance, Table 2 of the submitted paper shows that the new assembled model can achieve lower accuracy than

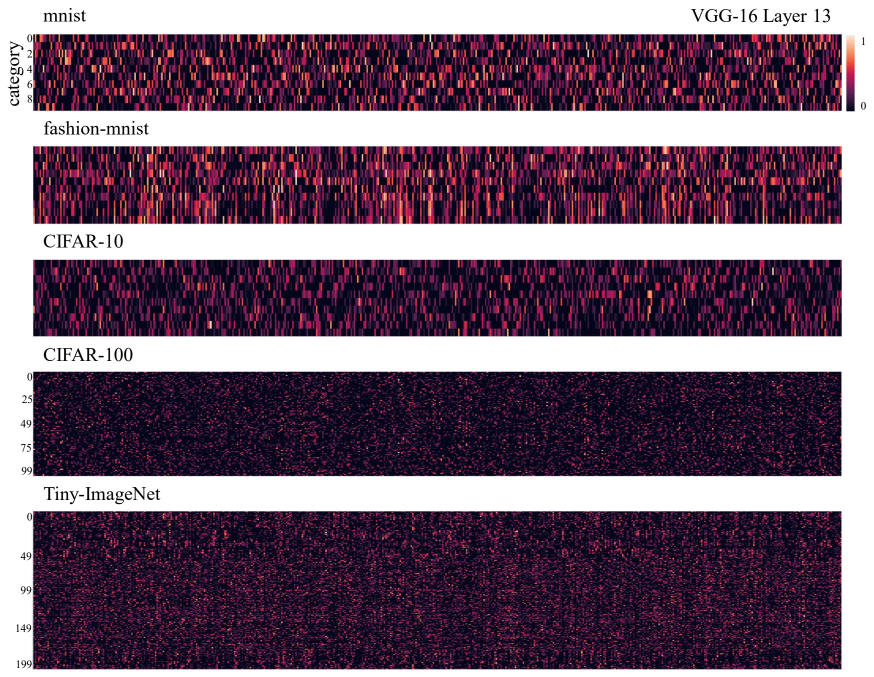

Figure 8: The total contribution patterns from different activation maps for samples of the different categories in layer 13 of VGG-16 on different datasets.

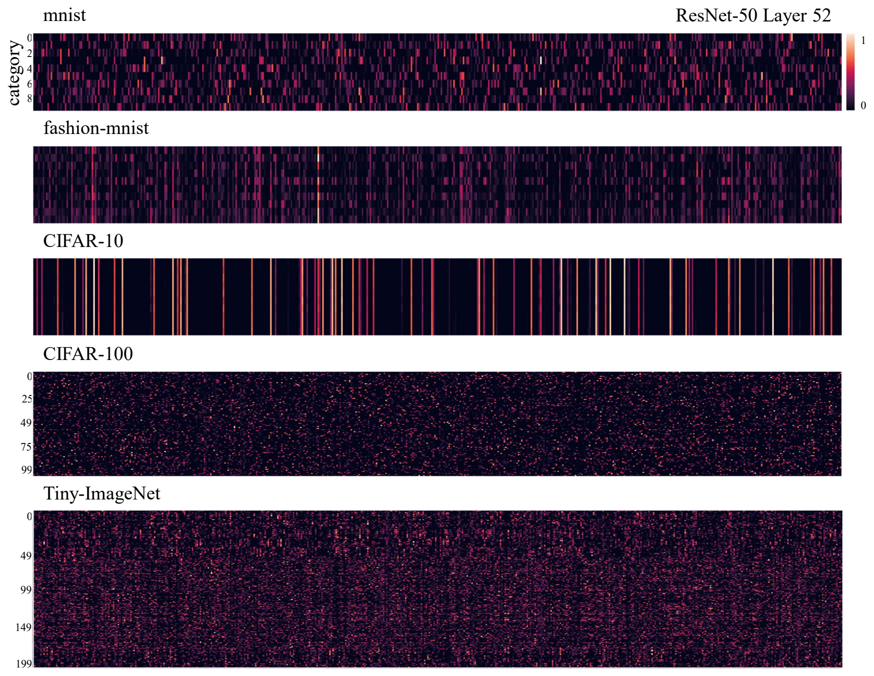

Figure 9: The total contribution patterns from different activation maps for samples of the different categories in layer 52 of ResNet-50 on different datasets.

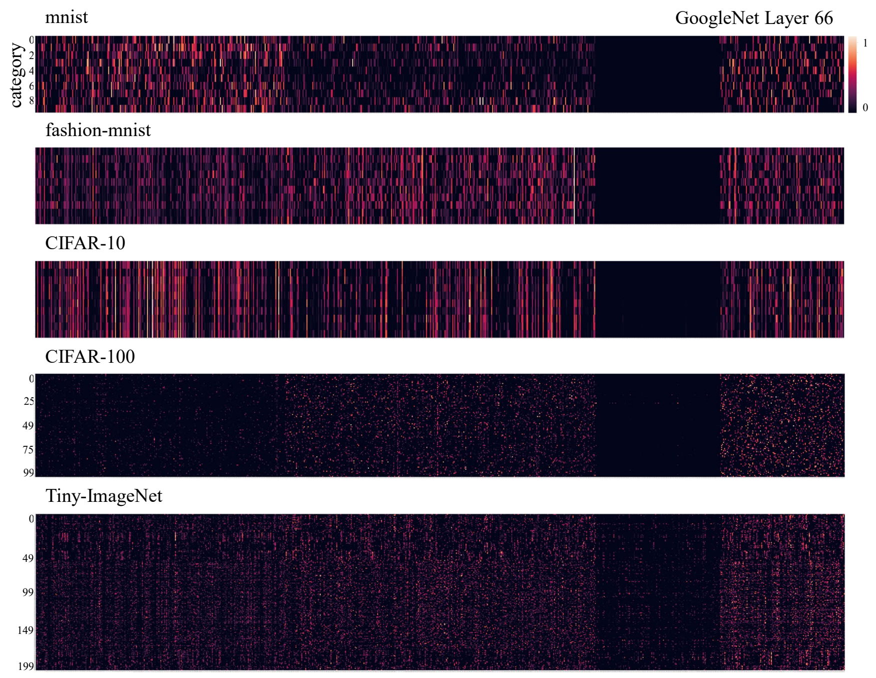

Figure 10: The total contribution patterns from different activation maps for samples of the different categories in layer 66 of GoogleNet on different datasets.

the original model on part cases such as assembling too many categories from different models (case ‘0-19 + 0-69’, case ‘20-69 + 70-179’, case ‘0-99 + 0-199’ of ‘CIFAR-100 + Tiny-ImageNet’). The possible reason is that more irrelevant components will disturb the correct prediction of samples. What’s more, for extensive applications, including model compression and knowledge distillation, the proposed model disassembling and assembling technique achieves poor performance than SOTA methods on part cases, which indicates that the proposed model disassembling and assembling technique needs further optimization on those tasks. Therefore, we will focus on solving the disturbance of irrelevant components and optimization of model disassembling and assembling for specific tasks in the future. In this paper, we only disassemble and assemble CNN classifiers. We will conduct disassembling and assembling models for other tasks such as object detection and segmentation in the future.