IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 27, 2020, accepted August 31, 2020, date of publication September 18, 2020, date of current version October 2, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3024959

Understanding a Deep Neural Network Based on Neural-Path Coding

DAWEI DAI[1,2], CHENGFU TANG[1], DONGYAO ZHAO[1], SHUYIN XIA[1], AND HAOYUE BAI[1]
[1]College of Computer Science and Technology, Chongqing University of Posts and Telecommunications, Chongqing 400065, China
[2]Department of Computer Science and Technology, Fudan University, Shanghai 200433, China

Corresponding author: Dawei Dai (daidw@cqupt.edu.cn)

This work was supported in part by the Natural Science Foundation of Chongqing under Grant E021D2019034, in part by the Chongqing Education Commission under Grant E010J2019025, and in part by the Natural Science Foundation of China Project under Grant 61936001 and Grant 61772096.

ABSTRACT Recently, the deep neural network has achieved considerable success in the field of machine learning. However, for most tasks, the current neural network models are still considered as the ''black-box'' structure, essentially because the knowledge a neural network learns from data is often unpredictable and unexplainable. Similar to neural coding in the brain, one neuron may be simultaneously involved in encoding one or even several tasks. Information may be transmitted between neurons along a particular neural-path (neural circuit) to end neurons that encode a specific decision. Inspired by this, we aim to explore the reason for the performance of a neural model based on a neural-path. First, for a trained neural model, we quantify the neural-path in which the role of a neuron was assumed to control the amount of information that can be passed through; Second, we define a Euclidean distance (ED) for every two neural-paths, and by analyzing the EDs between two classes of a neural classifier, we explain the ease of prediction for some classes, whereas others are not. We performed extensive experiments for architectures of ResNet and DenseNet models on several benchmark datasets, and determined that the shorter the distance between the neural-paths of two classes, the easier it is to make mistakes. Finally, we proposed a method for controlling the formation of a ''neural-path'' to build a partially understandable neural model. For the new ResNet model, each feature map in redirect layers was assigned to participate in encoding only one class. The feasibility of the method is also verified through experiments.

INDEX TERMS Interpretable neural network, neural-path, ResNet, DenseNet.

I. INTRODUCTION

Deep neural network (DNN) has become one of the important basic theories in several fields [4], [5], such as artificial intelligence, big data, and machine learning, because of its strong ability of learning and knowledge expression. In recent years, with the breakthrough of DNN models in the field of image processing, the application of DNN has experienced an explosive growth [6]–[8]. These DNN models have been deployed to a variety of applications such as automatic driving, cancer detection, recognition system, and complex games, and have reached or exceeded the level of human beings in several scenarios. Although the DNN model achieves superior performance in several fields, only a bunch of seemingly meaningless model parameters and very high fitting judgment results can be obtained.

Despite this progress, DNNs are still expected to make greater progress in theory to improve their comprehension. There remains no significant insight into DNN internal operation and behavior or its ability to achieve such a performance. For most tasks, the current DNN models are still considered as a ''black box'' structure. Unexplainable also means danger. In fact, in several fields, there are concerns about the application of deep learning model, not only the model itself cannot give sufficient information but also more or less about security. For example, application systems constructed near AI will often be involved and affect several fields, such as redefine medical interventions, autonomous transportation, criminal justice, financial risk management, and several other areas of society. However, considering the above challenges

The associate editor coordinating the review of this manuscript and approving it for publication was Wei Zhang.

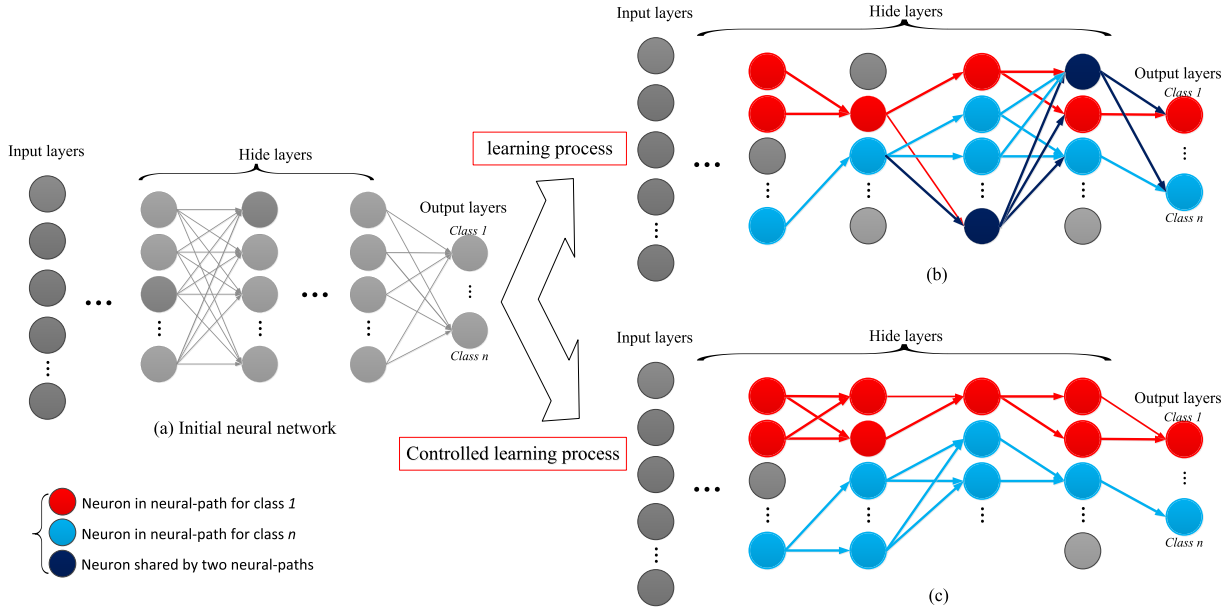

FIGURE 1. **Overview of neural-path in a deep neural network. (a) Initial neural network; (b). We hypothesis that the learning process is to learn the knowledge that we called "neural-path" in the image recognition task, which information flow of different classes may be transmitted along a particular "neural-path" from neurons in the input layer to some specific neurons in the output layer; (c). We proposed a method to control the learning process to obtain the particular "neural-path" that we expected.**

highlighted, the usefulness, fairness, and security of these AI systems will be gated by our ability to understand, explain, and control them. Consequently, it is becoming increasingly important to construct an understandable and interpretable neural network model. *The fundamental issues why we do not understand the neural models are noted in our manuscript as follows:*

- *Q1. We generally do not understand and even do not know what knowledge a neural network learns in a task.*
- *Q2. The knowledge learned by a neural network is unpredictable; generally, we just passively accept what the model has learned.*
- *Q3. The decision made by a neural model is generally not evaluable, i.e., For example, why did the agent do that and not something else? When does the agent succeed and when does it fail? When can I trust the agent?*

We can explain or understand the role of neurons (activation function) from different perspectives. For example, from the mathematical perspective, neurons can perform the role of nonlinear mapping, which makes the neural network model have more powerful learning ability; from the perspective of neural information coding, different information was selectively passed through a neuron in the brain in which information may be transmitted from start neurons along a particular ''neural-path'' from start to end that encode a specific decision. In the field of neuroscience, a substantial proportion of studies have been conducted on the structural details of a neural circuit (analogy to ''neural-path''), which is the key point to reveal the working mechanism of the nervous system [3], [9].

Inspired by the above neural coding, we hypothesize that a ''neural-path'' also exists in a DNN and encodes one class or a decision. The learning of a DNN can be regarded as the formation process of the neural-path, i.e., ''neural-path'' was regarded as the knowledge that a DNN learned in a task. Whether we can find and even control the formation of each neural-path in the learning of a DNN is meaningful to understand the working mechanism of a neural model. In one sense, each ''neural-path'' can be regarded as one standard template that is expressed a class. When a sample being calculated by a neural model can generate a current ''neural-path,'' by analyzing the matching degree of current and standard ''neural-path,'' we may explore the performance of a neural model.

In this study, we aim to test and verify the validity of this neural-path hypothesis. We define the concept of a ''neural-path'' as what explores and explains the decision-making of a neural network, regarded as the knowledge learned by a neural network model in a task, as illustrated in FIGURE 1. For a RELU based feed-forward neural network [10], only a portion of the activated neurons are involved in encoding one type of information in each layer. These neurons, residing in hierarchical layers, form a *neural-path*, which we assume encodes the information of a class. First, for a trained neural network model, we defined that if the average activation frequency of a neural node (neuron or a map) for all samples from the same class were greater than a threshold, the node was considered involved in encoding this class. Each *neural-path* encodes one class consisting of different nodes in each layer. Next, we defined an ED for every two neural-paths. Analyzing the distances between each two neural-paths of a

trained ANN classifier explains the performance of models. Finally, we proposed a method to control the formation of a ''neural-path'' in the learning process to construct an understandable model.

II. RELARED WORKS

In recent years, the construction of an interpretable DNN model has been garnering increasing attention. In this study, we divide the related work into two parts as follows: 1. Research on the interpretability of visualization method; that is, the hidden layer of neural network is transformed into an image that we can understand and visualize using certain methods; 2. The interpretability research based on the model; that is, using the understandable agent model or method to approach the DNN as follows.

A. VISUALIZATION METHODS

A very intuitive way to understand the concepts and features that are learned in each layer of neural network is to use some visualization methods to transform the hidden layer into an image with practical meaning that can be understood by human beings to help us intuitively understand the learning process or its working mechanism of deep learning [1], [2]. For example, Ramprasaath *et al.* generated a thermal map of the gradient of the convolution layer and visualized it to display and label the important pixels in the input image, which helps us understand the learning area of the DNN [11], [12]; Yosinski *et al.* visualized the knowledge learned by neural network by identifying the most active neurons in the input image [13], [14]; another type of method is the visualization of the significance or importance of the features—this method uses standard back propagation to calculate the partial derivative of the class score relative to the input features to measure the sensitivity of the classification score to the small change in the pixel value [15], [16]; Li *et al.* attempted to minimize the nonconvex loss function by visualizing DNN to determine the effect of the architecture and parameters on the loss situation [17]. These visualization methods make people have a direct impression on the internal mechanism of the DNN. However, because such methods cannot understand the internal structure of the model, it is difficult to directly explain the decision-making logic of the model. Therefore, the black box problem of the neural network has not been substantially solved.

B. MODEL DIAGNOSIS

There are two types of interpretability analysis based on model: agent model and automatic feature extraction. The method of proxy model involves constructing a new model to simulate the input and output of the black box model, and to understand the original black box model. A linear agent model for the description of model independent local interpretability was proposed by [18]–[20]; Zhang *et al.* added loss to filters in the convolution layer to obtain the interpretable semantic representation in the convolution layer, and they also proposed a prediction logic for quantitative interpretation of convolution network through a decision tree [21], [22]. The works [23], [24] sought to interpret neural networks from the perspective of geometry. Another field [25] is the brain-inspired models, which aim to design a biologically interpretable model. Automatic feature extraction is another method of DNN interpretability research. For example, KT method uses if-then rules to automatically extract features of each layer and neuron, and sensitivity analysis method to determine the importance of input variables through connection weight, deflection, and change of input variables. Studies [26] on the types of features neural networks mainly rely on in image recognition tasks, and the corresponding results are strongly biased towards recognizing textures rather than shapes, which is in stark contrast to human behavioral evidence and reveals fundamentally different classification strategies. Although these methods can analyze the decision logic of the existing neural network to some extent, they cannot directly build an interpretable neural network.

C. SUMMARY

The above study can reveal whether the network is well trained (by analyzing the feature maps in each layer) or the part of the input image that is sensitive to the neural network model or the features that have been learned by the neural model. This is important to understand the working mechanism of the neural network, but still not sufficient to interpret the neural network. For example, we still cannot understand the features learned by a neural model; we still cannot expect what neural models learned; and we still cannot interpret the performance of the models. Our contributions are summarized as follows:

(1) We hypothesize a concept "neural-path" as the knowledge that was learned using a neural model in a task, and we verified its effectiveness.

(2) Based on the matching degree of "neural-path" of different classes, we interpret the performance of a neural model. For example, "why are some classes easily predicted, while others are not?"

(3) We proposed a method to construct a partially understandable neural model by controlling the formation of the neural-path.

III. APPROACH

A. ABOUT NEURAL-PATH

In a well-trained neural classifier, each neuron can be considered as learning a type of feature. Lower layers appear concerned with low-level features such as color, edge, and texture. The higher the layer, the more abstract and complex the associated features are [27]. As we know, the features of samples from the same class are essentially similar. Therefore, when a trained neural network calculates different samples from the same class, the distribution of frequency of active neural nodes and non-active neurons in the network are assumed to be similar; that is, in a well-trained network

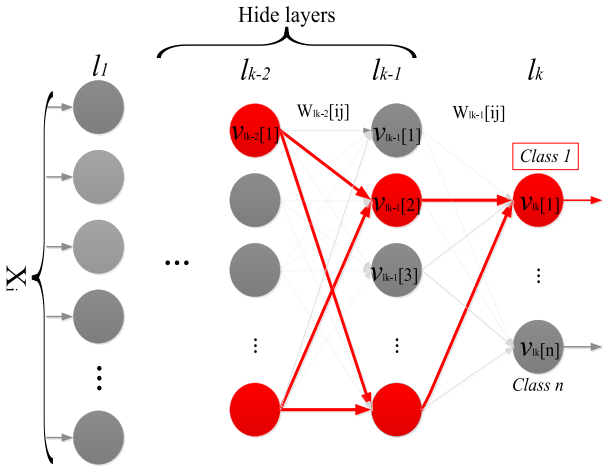

FIGURE 2. Diagrammatic sketch of a "Neural-path," the "neural-path" for class 1 was made up of the red neural nodes, which encoded the information of class 1; while the gray nodes were not participated in encoding the class 1.

model, samples from the same class have a high probability of taking the similar neural-path through the network, and eventually converge to an output neural node that represents this class label. Thus, in this study, a ''neural-path'' is essentially the flow direction of information in a hierarchical neural network.

B. DEFINE NEURAL-PATH

1) FULLY CONNECTED LAYER

FIGURE 2. illustrates a neural-path of a class consisting of red nodes in hierarchical layers. Gray nodes are not in this neural path. In once forward calculation for a specific class, nodes with zero or a small activation value offer no contribution to the nodes in the next layer, and are regarded as not participating in coding this class. Consequently, defining the neural-path for a specific class entails finding the ''red nodes,'' which participate in encoding this class. For a trained neural network model, we calculated the average activation of each neuron for all samples of one specific class. In each layer, we sorted all neural nodes according to their average activation value, and selected $\boldsymbol{m}$ nodes with the largest value as the nodes of this neural-path.

Taking one class as an example, (1), (2), (3) denote the quantification of neural-path.

$$V_{lk}[i] = \frac{1}{n}\sum_{j=0}^{n} v_{lk}[j] \tag{1}$$

$$Nv_{lk} : \left\{ V_{lk}[1],\ V_{lk}[2],\ \cdots\ V_{lk}[i],\ \cdots\ V_{lk}[m] \right\} \tag{2}$$

$$np_{class[1]} : \{\cdots\ Nv[h],\ Nv[h+1],\ \cdots\} \tag{3}$$

Term $V_{lk}[i]$ denotes the average node value of the i^{th} node in the k^{th} layer in which v_{lk} denotes the node value (*activation value*) for the sample X_j. Term $\boldsymbol{n}$ denotes the number of samples of $class_{[1]}$. Additionally, the value of one neural node was regarded as the amount of information that can pass through this neuron. Term Nv_{lk} denotes the response of the k^{th} layer that we selected; $np_{class[1]}$ indicates the neural-path for class 1.

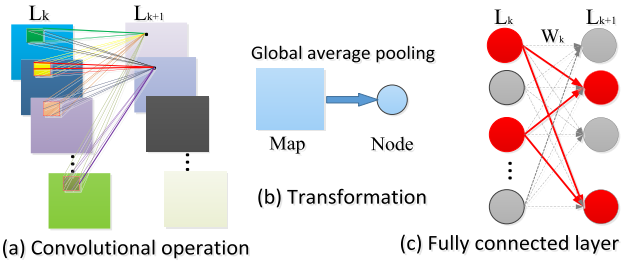

FIGURE 3. The procedure of convolutional operation to full connection.

2) CONVOLUTIONAL LAYER

The two main advantages of the convolutional layer as compared to the fully connected layer are local connections and weight sharing, typically operated inter *one feature map*. If one feature map was regarded as a node (global average or sum pooling a feature map), the convolutional layer transformed into a fully connected layer as illustrated in FIGURE 3. Thus, we can obtain node values in fully connected layers according to the previous section.

Convolution kernels in lower layers appeared involved in low-level features such as color, edge, and texture, whereas those in higher layers involve increasingly abstract and complex features. For any natural image, the lower the feature, the more similar it is [27]. We consider that distinguishing an object is mainly based on high-level characteristics. Consequently, we defined the neural-path as initiating from middle and higher convolutional layers.

C. DEFINING THE DISTANCE OF NEURAL-PATH

A trained n-classifier model typically has n output neurons. Ideally, the input sample from class-i sets the activation value of the i^{th} neuron in the output layer to 1 and others to 0. The previous section demonstrated that we could obtain a neural-path that includes a set of node values in the hierarchical layers for each class, according to (1), (2), (3). The information of each class only passes through neural nodes with positive node values in a neural-path. In this way, we can obtain n neural-path and $n(n\text{-}1)$ EDs in an n-classifier neural model, which can encode the information of each class in samples respectively. We define ED of the i^{th} and j^{th} classes based on their neural-paths, as defined in (4).

$$\begin{aligned} ||D_{ij}||_2 &= (NP_i - NP_j)^2 \\ &= \frac{1}{n - L_k}\sum_{L:l=L_k}^{n}\frac{1}{N_l}\sum_{N:m=0}^{N_l}(N_{Ll\_i}[m] - N_{Ll\_j}[m])^2 \end{aligned} \tag{4}$$

Term $||D_{ij}||_2$ quantitatively represents the similarity of the i^{th} and j^{th} classes. Intuitively, the smaller the ED $||D_{ij}||_2$ between the i^{th} class and j^{th} class (where $||D_{ij}||_2 < ||D_{ik}||_2, j! = k$), the greater the probability that the i^{th} class is predicted as the j^{th} class. If $||D_{ih}||_2$ between the i^{th} class and

other classes (where $h = 1, 2, \ldots, n$) are clearly smaller than $||D_{jh}||_2$ (where $h = 1, 2, \ldots, n, i \; ! = j$), the trained model should perform more poorly for the i^{th} class samples than for those of the j^{th} class.

D. CONTROLLING THE FORMATION PROCESS OF NEURAL-PATH

In the above sections, we defined "neural-path" as the knowledge that a neural network learned, and we can regard the learning process of a neural network as the formation of a neural-path for each class in a classification task. However, such a neural-path obtained through learning was unpredictable in a traditional way, and we usually passively accepted such information learned by a neural model, which was considered as one of the main reasons for their unexplainable. Herein, we attempt to design a neural network with a predictable neural-path in a classification task.

For a CNN, our approach is to assign only specific feature maps $X_{\text{part}-0}$ in a selected layer to encode a particular class, it is 0 herein, whereas other maps ($X_{\text{part}-1}$) in this layer encode class 1, as illustrated in FIGURE 5.(a). We can achieve this target by designing special loss functions, as (5), (6), (7). $\text{Loss}_{\text{pass}}(x_0)$: mean of sum of squares of feature maps $X_{\text{part}-0}$ assigned approaches a constant c for all samples of class 0, indicating that only $X_{\text{part}-0}$ allows information of class 0 to pass through. $\text{Loss}_{\text{forbid}}(x_0)$: squares of feature maps $X_{\text{part}-1}$ assigned approaches 0, indicating that feature maps $X_{\text{part}-1}$ forbid information of class 0 to pass through. $\text{Loss}_{\text{redirect}}$ is the weighted sum of $\text{loss}_{\text{pass}}$ and $\text{loss}_{\text{forbid}}$, achieving only $X_{\text{part}-0}$ encoding class 0. $\text{loss}_{\text{classification}}$ was cross entropy loss function for classification. For a classification task, final function can be expressed as (8) as follows

$$loss_{pass}(x_0) = \frac{1}{m}\sum_i (X_{part-0}[i] - c)^2 \tag{5}$$

$$loss_{forbid}(x_0) = \frac{1}{m}\sum_i (X_{part-1}[i])^2 \tag{6}$$

$$loss_{redirect} = loss_{pass} + loss_{forbid} \tag{7}$$

$$loss = \alpha \cdot loss_{redirect} + \beta \cdot loss_{classification} \tag{8}$$

IV. EXPERIMENTS AND ANALYSES

We demonstrated the effectiveness of the proposed method for MLP and CNN models on several benchmark datasets (including MNIST, Cifar10 [28], Cifar100, and ImageNet [29]). Cifar10 dataset consists of 50,000 training and 10,000 testing images from 10 classes; Cifar100 dataset consists of 60,000 natural images from 100 classes, each class contains 500 training images and 100 testing images; ImageNet32 dataset is a down-sampled version of the original ImageNet, and consists of 1.2 million images for training and 50,000 for validation from 1,000 classes. The classification task for the ImageNet32 dataset was more complex than that of the Cifar dataset. We performed the training experiments on the training dataset and evaluated the results using the test dataset.

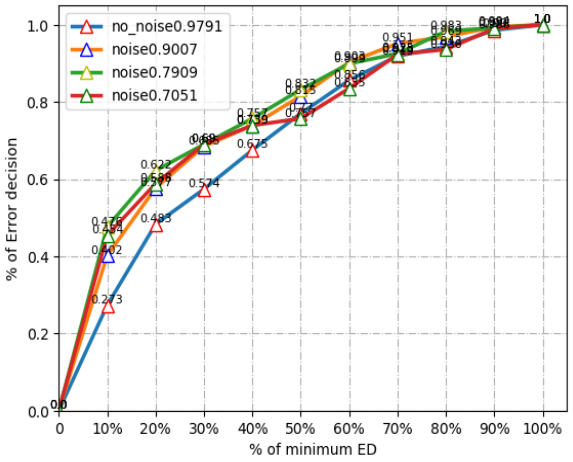

FIGURE 4. The smaller ED of each pair of neural-path will get more higher incorrect decision rate whatever samples with different levels of noise.

A. MLP MODEL ON MNIST

We first illustrate the proposed approach on a trained MLP model on the MNIST dataset. For the MLP classifier, we employ a structure of 784*400*400*400*10 with three hidden layers with 97.91 % to obtain a specific number of incorrect decisions, where the neuron models in the hidden layer are ReLU and end with Softmax layer.

We obtain the neural-path for each class based on (1), (2), (3). We first analyzed the relationship of EDs of each two neural-paths and their wrong decisions. FIGURE 6 illustrates each ED between one neural-path that encodes one class and others, and their corresponding incorrect decisions. The statistics note that the ED of two classes is approximately inversely proportional to the number of wrong decisions in these classes, i.e., the smaller the ED of two neural-paths, the more frequently the two classes encoded by these two neural-paths incorrectly predict each other. Consequently, the ED of each two neural-paths can be regarded as one type of measurement of their similarity, i.e., the shorter the distance between them, the more similar they are.

To obtain more test examples with incorrect decisions, we add different intensity of noises to the MNIST test dataset (with 97, 90, 80, and 70 %). From a statistical perspective, more samples will accurately exhibit such a rule. Here, we provided general statistics on samples where the neural model issued incorrect decisions. FIGURE 6 indicates that approximately the top 20% minimal EDs account for approximately 49.5–62% of all incorrectly predicted examples; the ED is indirectly proportional to the increase in the number of incorrect decisions. We can therefore determine that most incorrect decisions made by a neural model are more likely to occur in samples with a small ED of the neural-path. From the curves with different number of samples, we can note that such rules are clearer in tests with more samples.

B. CNN MODELS ON CIFAR-10

We often consider that the deeper the network, the stronger the expression ability. However, the experiments identified

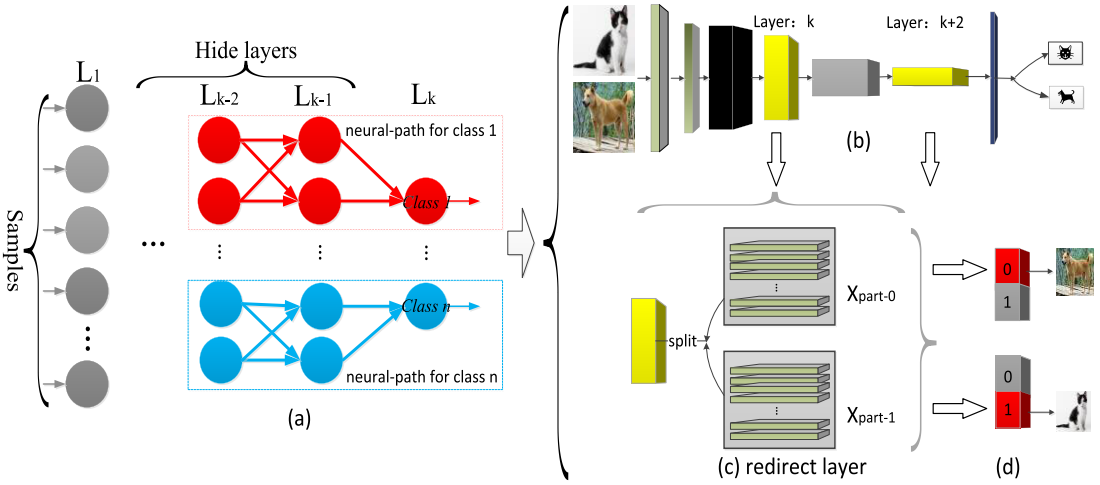

FIGURE 5. (a). Learning a desired neural-path for a DNN by assigning some particular neurons; For a CNN (2 classification tasks), step (b) ->(c)->(d), we first select a specific layer that we named "redirect layer", and split the feature maps into two parts, each part only participate in encoding one class (part 0 for dog, and part 1 for cat).

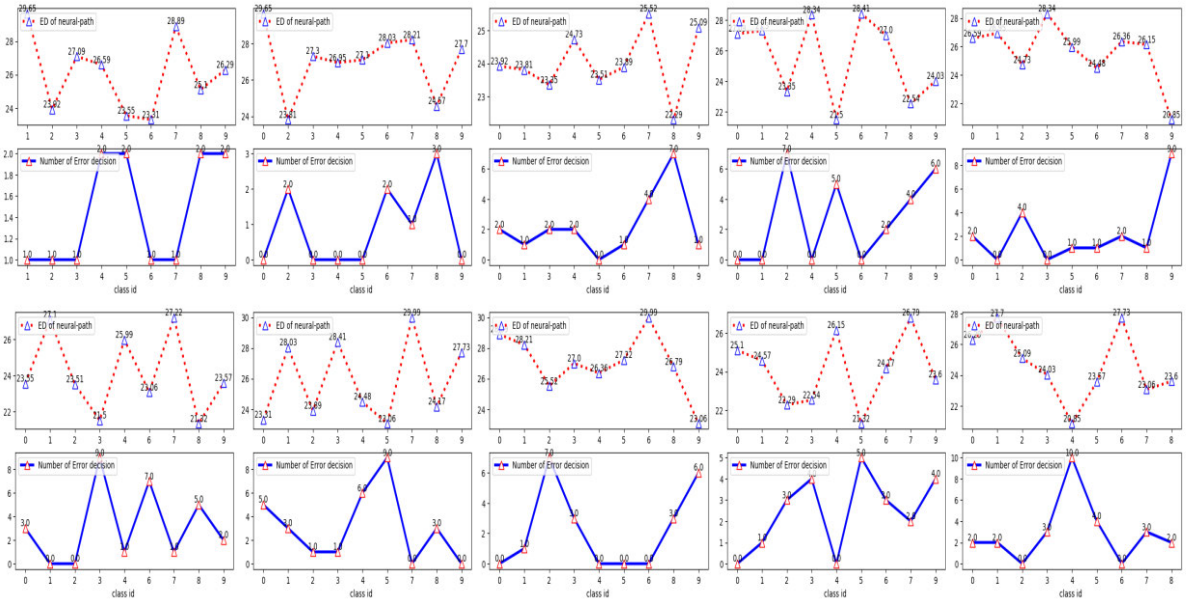

FIGURE 6. ED between each pair of neural-paths and corresponding number of incorrect decisions for an MLP model on MNIST, from classes 0 to 9. Upper graph of each sub-figure depicts EDs of one class with others, whereas lower graph depicts the number of wrong decisions on other classes.

two problems: gradient dispersion or disappearance, and performance degradation with network deepening [5], [31]. ResNet [32] alleviates the problem using a shortcut structure, and DenseNet [33] further improves the transmission efficiency of information and gradients in the network, now extensively used in several fields. Consequently, we demonstrated the effectiveness of the proposed method on ResNet and DenseNet models.

1) NETWORK ARCHITECTURE FOR RESNET AND DENSENET

We evaluated our method in the ResNet and DenseNet models on the CIFAR-10 dataset. For ResNet, the first layer was a 3×3 convolution kernel. Thereafter, we used a stack of 6*n layers with 3×3 convolution kernels on feature maps of sizes 32, 16, and 8, respectively, with 2*n layers for each feature map. The numbers of filters were 16, 32, and 64 respectively (16*k, 32*k, and 64*k respectively for Wide ResNet [34], where k is the coefficient). The sub-sampling was performed by convolutions with a stride of 2. For DenseNet, a convolutional layer with 24 output channels was first used with the input images. For convolutional layers with kernel size 1×1 and 3×3 (bottle block). We used 1×1 convolution followed by 2×2 average pooling as transition layers between two dense blocks. At the end of the last block (both ResNet and

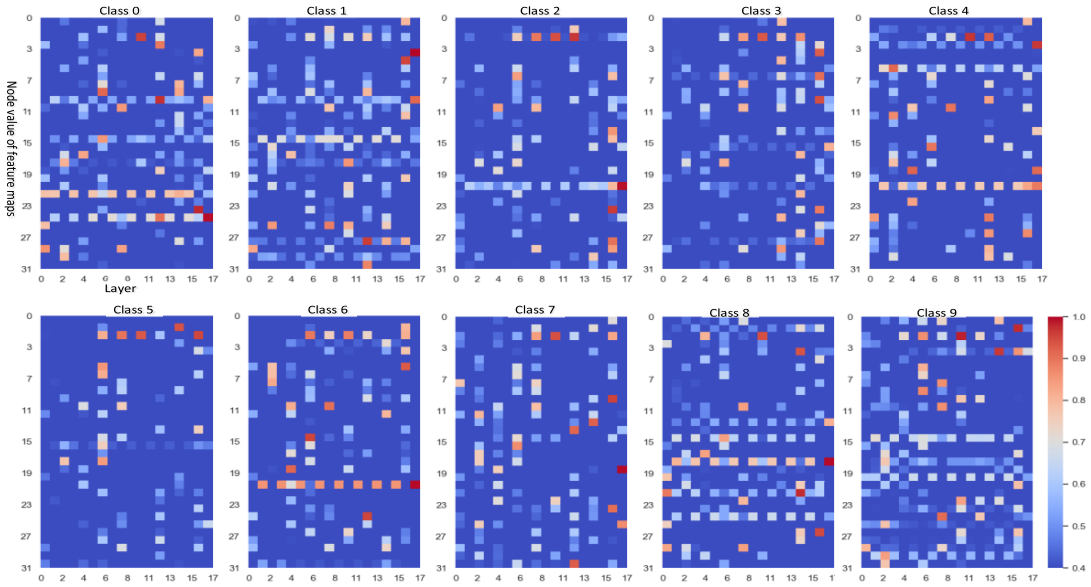

FIGURE 7. Visualization part of node value of feature maps of the last 18 layers of ResNet-56 for 10 classes.

DenseNet), a global average pooling was performed and then a Softmax classifier attached. The feature map sizes in the three blocks were 32 × 32, 16 × 16, and 8 × 8, respectively.

2) IMPLEMENTATION

This study aimed to verify our neural-path concept that coding the information that is learned by a neural network, but not to push the state-of-the-art results. Thus, we did not explore a specific skill in the training process, and thus, we only adopted similar implementations (including data preprocessing and training skill [32]) with the original models. For 20-layer (and 56-layer) ResNet and Wide ResNet models, feature maps of the last 10 (and 28) layers were selected to define the neural-path for each class. Feature maps of each layer were first passed through a batch normalization layer [35], regarded as a type of sparse. We summed the node activation values for all training samples to determine node values corresponding to the feature map, and obtained each neural-path and ED according to (1), (2), (3), (4).

3) RESULTS AND ANALYSES

We had trained the neural models (including ResNet-20, ResNet-56, WRN-20, and DenseNet-BC-40) with 91.13 %, 92.78 %, 93.02 %, and 92.73 % on test datasets respectively. We obtained the neural-path of each class based on (1), (2), (3) for ResNet and DenseNet models, as illustrated in Fig.7. This illustrates that the value of the same node for different classes is frequently different, indicating that one node participates in encoding different class information to different degrees. We consider the node with values close to zero as having no contribution to that class. Information from the previous layer flows along the neural-path.

First, we provided general statistics on samples where the neural model issued incorrect decisions. Fig.8 indicates that

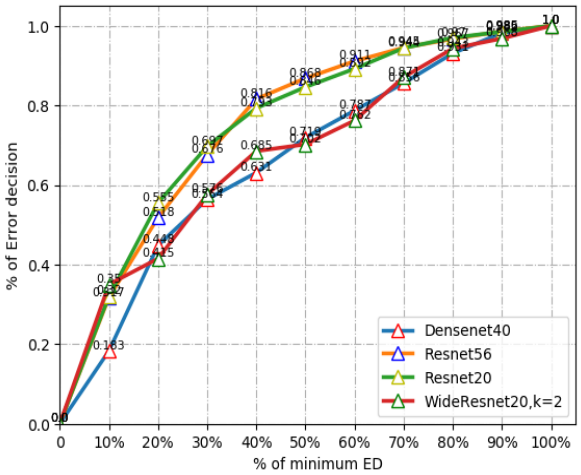

FIGURE 8. Incorrect decision for all EDs of each two neural paths.

the top 20 % (and top 30 %) minimal ED accounts for approximately 40–60 % (and 55–70 %) of all incorrectly predicted examples; the larger the ED, the smaller the increase in the number of incorrect decisions. We can therefore determine that most incorrect decision making for all neural models occurs in samples with a small neural-path. ED.

We further analyzed all EDs of each two neural paths on ResNet and DenseNet models. FIGURE 9 and FIGURE 10 illustrate each ED between one neural-path and others, and the corresponding incorrect decisions. The statistics indicate that the ED of two classes is inversely proportional to the number of wrong decisions in these classes, i.e., the smaller the ED of two neural-paths, the more frequently the two classes incorrectly predict each other. Therefore, the ED of each pair of neural-paths can be regarded as a proportional

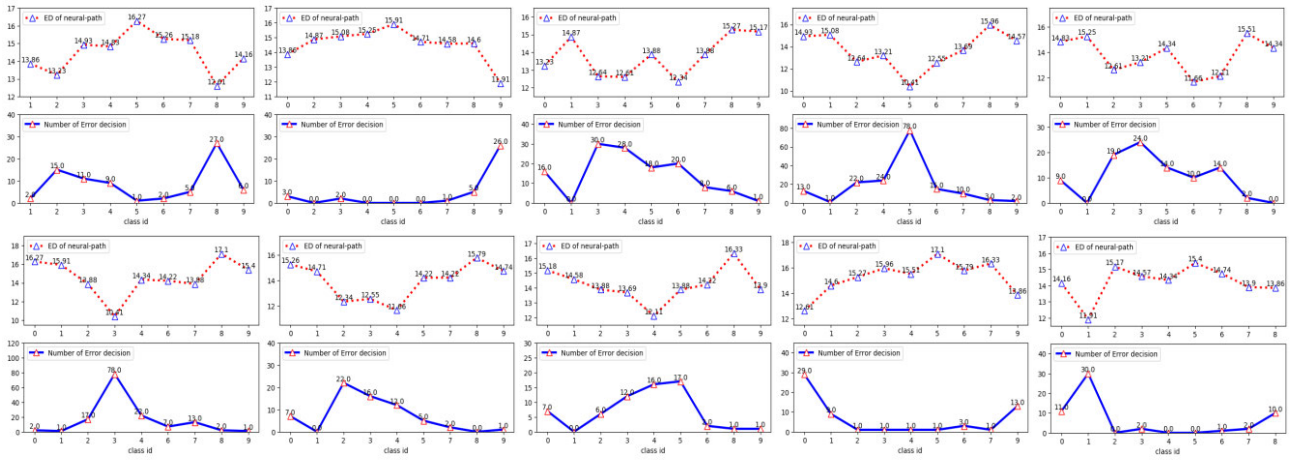

FIGURE 9. Statistics of ED between each pair of neural-paths, and their corresponding number of incorrect decision-makings for ResNet-20 model on Cifa10.

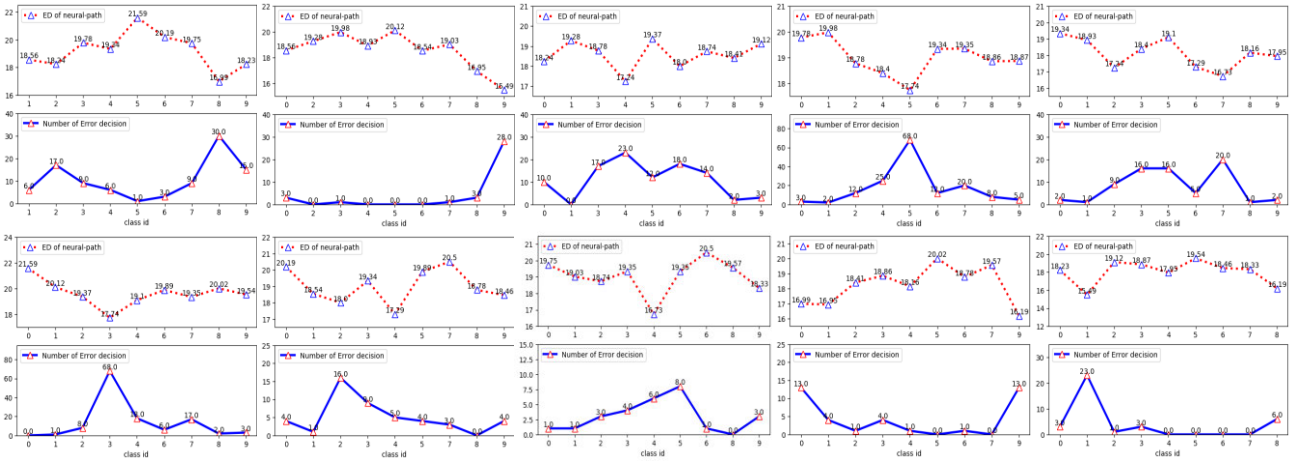

FIGURE 10. Statistics of ED between each pair of neural-paths, and their corresponding number of incorrect decision-makings for Densenet-40 model on Cifa10.

measurement of their similarity, i.e., the shorter the distance between them, the more similar they are.

From the curves of each class in FIGURE 9 and FIGURE 10 respectively, the two trends of the same class for two models are very similar, for example, if one class performs poorly on the ResNet model, and it is likely that it will not perform well on the DenseNet model, which also indicates an interesting phenomenon that these two models are likely to make similar mistakes for the same dataset.

C. CNN MODELS ON CIFAR100

1) NETWORK ARCHITECTURE AND IMPLEMENTATIONS

Section IV depicts the architecture of ResNet and DenseNet models. We only used the similar implementations of the original models. We selected feature maps of the last 10 layers (in the last block layer) to define the neural-path for each class. Other details are the same as in previous sections.

2) RESULTS AND ANALYSES

We trained the neural models ResNet-32, Wide ResNet-20, and DenseNet-40 with 75.54 % and 73.54 % on Cifar100 and 51.96 % on ImageNet32 datasets, respectively. We obtained the neural-path of each class from (1), (2), (3) for ResNet and DenseNet models. First, we provided general statistics on the incorrect decision samples of the neural model. From the statistics in FIGURE 11(a), the top 10 % (and top 20 %) minimal ED accounts for approximately 50–60 % (and 64–74 %) of all incorrectly predicted examples; the larger the ED, the smaller the increase in the number of incorrect decisions. The similar results can be obtained in FIGURE 11(b).

D. CONTROL THE FORMATION OF NEURAL-PATH

1) NETWORK ARCHITECTURE DESIGNING

We named the module of controlling the neural-path of information as the redirect module, and we called a convolution

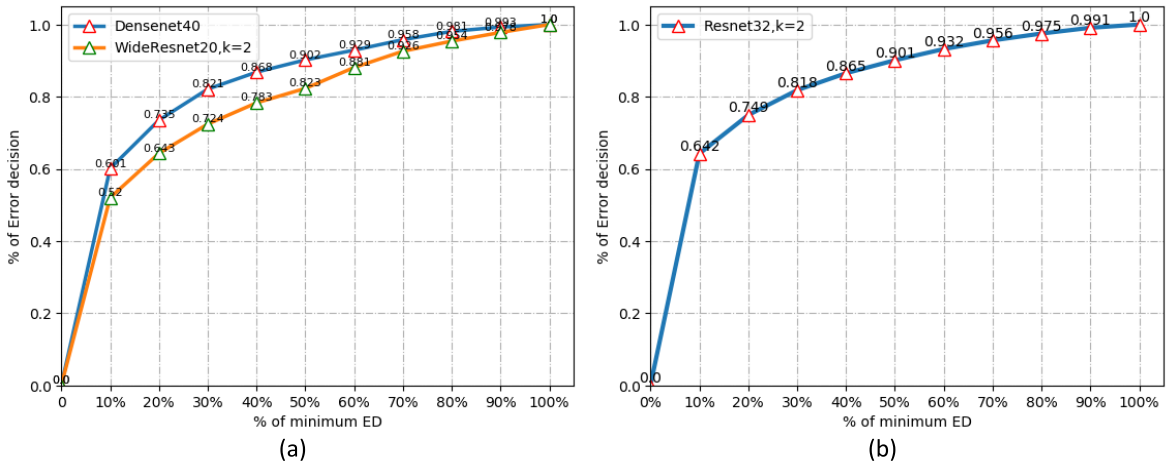

FIGURE 11. Overall statistics on the samples with incorrect decision of Resnet and Densenet. (a). on Cifar100 and (b). on Imagenet32.

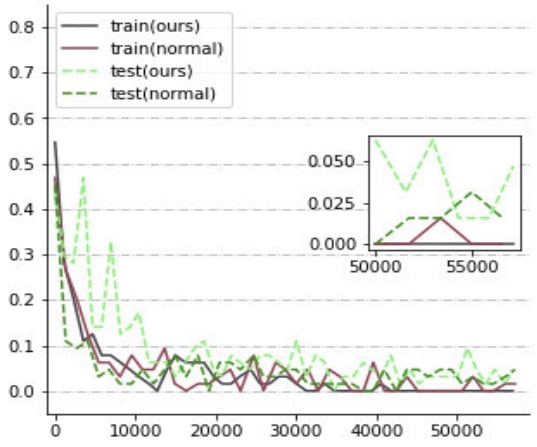

FIGURE 12. Training process of ResNet-50 with redirect modules and normal one.

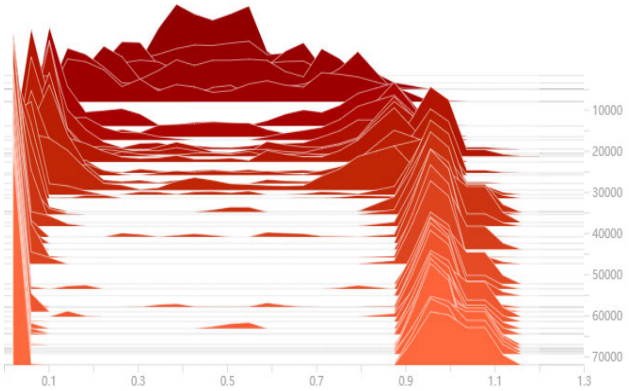

FIGURE 13. The amount of information that can be passed through the assigned neurons of the first redirect layer in the training process for the train dataset, horizontal axis notes the average node-value (Xpart-0, Xpart-1 as illustrated in FIGURE 3 (c)); Longitudinal axis notes the training step; Height of mountain note the number of sample of the current batch.

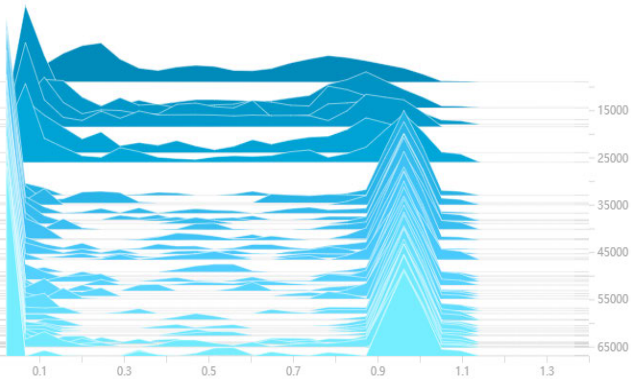

FIGURE 14. Similar exhibition in the training process for the validation dataset.

layer with a redirect module as the redirect layer. We selected two convolution layers in the third and fourth residual blocks as redirect layers. The details for controlling the formation of a neural-path of a 2-classification task, if the number of channels is 128, we assigned the half of feature maps to class 0 and the other half of feature maps to class 1 in first redirect module; similar operations were adopted in the second redirect module. Only feature maps that assigned to a specific class participate in encoding this class, others have no means to this class. In the construction of network structure, we used a 50-layer Residual Net as the basic network structure, but we decreased up to four times the filter numbers in each convolution because the experimental dataset can only include two classes. Architecture designing was summarized in TABLE 1.

2) IMPLEMENTATION

We select dogs vs. cats dataset in Kaggle to verify this idea. The dataset consisted of 25,000 images including "dog" and "cat". Each class contains 12,500 training images. We split the dataset into 20,000 training and 5,000 validation images. We resize by randomly cropping training images size to [224,224] and the horizontal flip and rotation image augmen-

TABLE 1. Network architecture for Residual Net with redirect layers.

Layer	Out size	Architecture	Reallocation
conv1	112×112	7×7, 16, stride 2	
Residual blocks 1	56×56	3×3 max pool, stride 2	
		$\begin{bmatrix} 3\times3, 16 \\ 3\times3, 16 \end{bmatrix}\times3$	
Residual blocks 2	28×28	$\begin{bmatrix} 3\times3, 32 \\ 3\times3, 32 \end{bmatrix}\times4$	
Residual blocks 3 with redirect module	14×14	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times6$	The feature maps [1:32] of the last layer in this residual block only participate in encoding class 0, the others [33:64] encode class 1.
Residual blocks 4 with redirect module	7×7	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times3$	The feature maps [1:64] of the last layer in this residual block only participate in encoding class 0, the others [65:128] encode class 1.
	1×1	global average pool, 2-d fc, softmax	

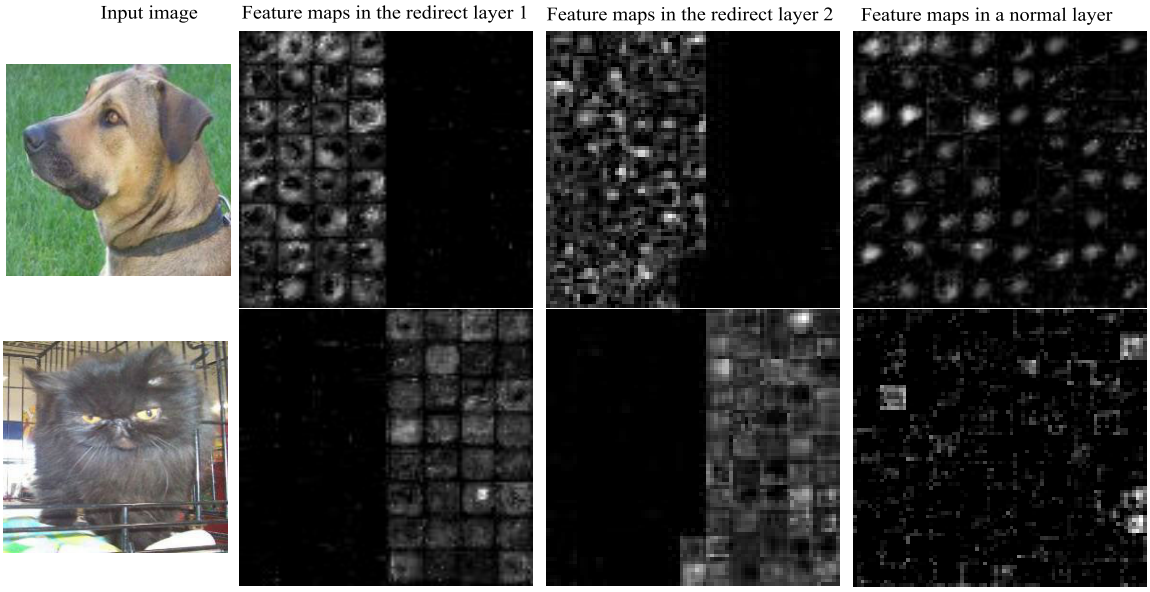

FIGURE 15. Visualization of part of feature maps in redirect layers and normal layers. For the samples with the correct decision, in the redirect layer, only the preassigned nodes participated in encoding the information, which was not obvious in a normal layer.

tation is in use. Thereafter, dividing images by 255 to provide them in range [0,1] as input. We use Adam optimizer [35] to train the model with mini-batch size of 64 and total training iteration is 58000.

3) RESULT AND ANALYSIS

FIGURE 12 illustrates the training and validation accuracy curves of the Resnet-50 with and without redirect modules in the training procedure. We noted that the model with redirect modules exhibited a lower training accuracy and almost similar to the original model on the train datasets. The new model can also obtain a very close accuracy on the validation dataset compared to the original model (97.92 % vs 98.02 %). Consequently, the learning and generalization abilities of such Resnet-50 with redirect modules have not declined significantly.

FIGURE 13 and FIGURE 14 illustrate the controlling process of the neural-path in the one redirect layer on train and validation datasets respectively. Average node-value of feature maps in redirect layers indicates the amount of information that can be passed through. From the training process, initially, all the maps (or neurons) participated in encoding the information in the form of

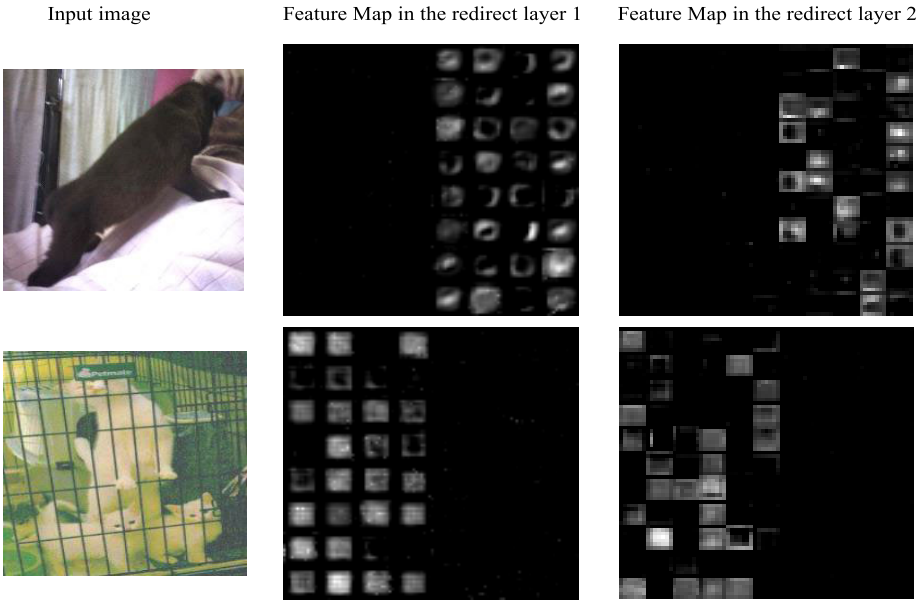

FIGURE 16. Visualization of part of feature maps in the two redirect layers for the samples that with incorrect decision. For example, the image of "dog" was mistakenly identified as "cat", and "cat" was mistakenly identified as "dog", from the redirect layers, we can observe that the nodes assigned to "cat" ("dog") participated in encoding the "dog" ("cat"), leading the incorrect decisions.

normal distribution. As the learning process continues, to the greatest extent, information (increasing samples) is allowed to pass or not allowed to pass, which is characterized by two node values one close to 1 and another close to 0.

For two specific examples as illustrated in FIGURE 15, we visualized the feature maps of a normal layer and a redirect layer in the trained ResNet-50 model in which we can observe that only particular feature maps in the redirect layer were activated by only the samples from one class, i.e., only the preassigned nodes encodes the information of a specific class. Meanwhile, in a normal layer (non-redirect layer), we can observe that almost all the nodes more or less participate in information coding. For the samples with incorrect decision, as illustrated in FIGURE 16, some nodes that are not assigned for this class more or less participated in encoding such a class, i.e., the information was not transmitted along with the preassigned ''neural-path'', leading to the model making mistakes.

Through experiments, we can assign some specific neurons to encode a particular class. Consequently, we can hypothesize that the information of examples with correct decision-making is most likely to go through the pre assigned nodes in the redirect layer in which the node-value of the assigned maps is close to 1 and others are close to 0; Conversely, if information flows far away from 1 or 0, for example close to 0.5, the decision-making is more likely to be untrustworthy.

V. DISCUSSION AND FUTURE WORKS

In summary, first, we assume that a neural model can encode information through the ''neural-path'' as what the neural network learned in a task in which we considered neurons to control the amount information that can be passed through. We also quantify the *neural-path* and make further analysis for ResNet and DenseNet models on several benchmark datasets; Second, we attempt to construct a partially understandable and expectable neural model based on a *neural-path*, and we proposed a method to control the formation of a neural-path by assigning only the expected neurons to encode the particular information.

The experimental results supported this hypothesis significantly. From the results, ***first,*** we note that if the samples from class-i are predicted incorrectly compared to other classes, which were more likely to be the several specific classes that with small neural-paths' ED, we attempt to answer question Q3: why some classes are easily predicted, while others are not?; ***Second***, we also observe an interesting phenomenon in which a neural model was likely to make similar mistakes for some dataset; ***Finally***, we constructed a new model based on baseline ResNet model, which performs almost as well as the baseline model; however, only the expected nodes of the new model participated in encoding the preassigned information, and this new model was partially understandable and expectable for us.

In future studies, we would focus on building the understandable neural models by designing expectable neural-path, and try to explore the following: when does the agent succeed and when does it **fail?** and when can **I** trust the **agent?**

REFERENCES

[1] M. D. Zeiler and R. Fergus, ''Visualizing and understanding convolutional networks,'' in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2014, pp. 818–833.

[2] M. D. Zeiler, G. W. Taylor, and R. Fergus, "Adaptive deconvolutional networks for mid and high level feature learning," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 2018–2025.
[3] H. Wei, D. Dai, and Y. Bu, "A plausible neural circuit for decision making and its formation based on reinforcement learning," *Cognit. Neurodyn.*, vol. 11, no. 3, pp. 259–281, Jun. 2017.
[4] M. Rahman Minar and J. Naher, "Recent advances in deep learning: An overview," 2018, *arXiv:1807.08169*. [Online]. Available: http://arxiv.org/abs/1807.08169
[5] S. Sun, W. Chen, L. Wang, X. Liu, and T.-Y. Liu, "On the depth of deep neural networks: A theoretical view," 2015, *arXiv:1506.05232*. [Online]. Available: http://arxiv.org/abs/1506.05232
[6] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep Learning*, vol. 1. Cambridge, MA, USA: MIT Press, 2016.
[7] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Deep learning for time series classification: A review," *Data Mining Knowl. Discovery*, vol. 33, no. 4, pp. 917–963, Jul. 2019.
[8] D. C. Cireçan, U. Meier, J. Masci, L. M. Gambardella, and J. Schmidhuber, "High-performance neural networks for visual object classification," 2011, *arXiv:1102.0183*. [Online]. Available: http://arxiv.org/abs/1102.0183
[9] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," 2016, *arXiv:1611.03530*. [Online]. Available: http://arxiv.org/abs/1611.03530
[10] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proc. 14th Int. Conf. Artif. Intell. Statist.*, 2011, pp. 315–323.
[11] R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra, "Grad-CAM: Why did you say that?" 2016, *arXiv:1611.07450*. [Online]. Available: http://arxiv.org/abs/1611.07450
[12] Z. Zhang, Y. Xie, F. Xing, M. McGough, and L. Yang, "MDNet: A semantically and visually interpretable medical image diagnosis network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6428–6436.
[13] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson, "Understanding neural networks through deep visualization," 2015, *arXiv:1506.06579*. [Online]. Available: http://arxiv.org/abs/1506.06579
[14] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 487–495.
[15] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," 2014, *arXiv:1405.3531*. [Online]. Available: http://arxiv.org/abs/1405.3531
[16] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," 2017, *arXiv:1704.02685*. [Online]. Available: http://arxiv.org/abs/1704.02685
[17] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein, "Visualizing the loss landscape of neural nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 6389–6399.
[18] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should I trust you?': Explaining the predictions of any classifier," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2016, pp. 1135–1144.
[19] R. Hu, J. Andreas, T. Darrell, and K. Saenko, "Explainable neural computation via stack neural module networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 53–69.
[20] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 4765–4774.
[21] Q. Zhang, Y. N. Wu, and S.-C. Zhu, "Interpretable convolutional neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8827–8836.
[22] Q. Zhang, R. Cao, F. Shi, Y. Nian Wu, and S.-C. Zhu, "Interpreting CNN knowledge via an explanatory graph," 2017, *arXiv:1708.01785*. [Online]. Available: http://arxiv.org/abs/1708.01785
[23] N. Lei, D. An, Y. Guo, K. Su, S. Liu, Z. Luo, S.-T. Yau, and X. Gu, "A geometric understanding of deep learning," *Engineering*, vol. 6, no. 3, pp. 361–374, Mar. 2020.
[24] X. Gu, F. Luo, J. Sun, and S.-T. Yau, "Variational principles for minkowski type problems, discrete optimal transport, and discrete monge-ampere equations," 2013, *arXiv:1302.5472*. [Online]. Available: http://arxiv.org/abs/1302.5472
[25] N. Kriegeskorte and P. K. Douglas, "Cognitive computational neuroscience," *Nature Neurosci.*, vol. 21, no. 9, pp. 1148–1160, 2018.
[26] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, and W. Brendel, "ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness," 2018, *arXiv:1811.12231*. [Online]. Available: http://arxiv.org/abs/1811.12231
[27] A. Krizhevsky, "Learning multiple layers of features from tiny images," M.S. thesis, Dept. Comput. Sci., Univ. Toronto, Toronto, ON, Canada, 2009.
[28] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.
[29] R. Sun, "Optimization for deep learning: Theory and algorithms," 2019, *arXiv:1912.08957*. [Online]. Available: http://arxiv.org/abs/1912.08957
[30] S. Sun, W. Chen, L. Wang, X. Liu, and T.-Y. Liu, "On the depth of deep neural networks: A theoretical view," 2015, *arXiv:1506.05232*. [Online]. Available: http://arxiv.org/abs/1506.05232
[31] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[32] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4700–4708.
[33] S. Zagoruyko and N. Komodakis, "Wide residual networks," 2016, *arXiv:1605.07146*. [Online]. Available: http://arxiv.org/abs/1605.07146
[34] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*. [Online]. Available: http://arxiv.org/abs/1502.03167
[35] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980