Neural Computing and Applications (2023) 35:10801–10819
https://doi.org/10.1007/s00521-023-08265-x

ORIGINAL ARTICLE

Identification of untrained class data using neuron clusters

Young-Woo Lee[1] · Heung-Seok Chae[1]

Received: 17 May 2022 / Accepted: 6 January 2023 / Published online: 31 January 2023

Abstract

Convolutional neural networks (CNNs), a representative type of deep neural networks, are used in various fields. There are problems that should be solved to operate CNN in the real-world. In real-world operating environments, the CNN's performance may be degraded due to data of untrained types, which limits its operability. In this study, we propose a method for identifying data of a type that the model has not trained on based on the neuron cluster, a set of neurons activated based on the type of input data. In experiments performed on the ResNet model with the MNIST, CIFAR-10, and STL-10 datasets, the proposed method identifies data of untrained and trained types with an accuracy of 85% or higher. The more data used for neuron cluster identification, the higher the accuracy; conversely, the more complex the dataset's characteristics, the lower the accuracy. The proposed method uses only the information of activated neurons without any addition or modification of the model's structure; hence, the computational cost is low without affecting the classification performance of the model.

Keywords Neuron cluster · Similarity analysis · Identifiability · Untrained class data · Robustness

1 Introduction

Deep learning has been studied since 1940 and has recently attracted attention with the development of technology and hardware for artificial neural networks [1]. In addition, artificial neural networks that can handle large amounts of data and solve complex problems are required. Deep neural networks (DNNs), which increase the number of hidden layers in artificial neural networks, have emerged to meet this requirement [2].

Convolutional neural networks (CNNs) are a representative type of DNN and are used in various fields, such as image classification [3, 4], face recognition [5], and video processing [6], as well as in safety–critical systems [7–9], such as those of autonomous vehicles, in which robustness is very important [10–12].

✉ Heung-Seok Chae
hschae@pusan.ac.kr

Young-Woo Lee
amorepooh@pusan.ac.kr

[1] Department of Electrical & Computer Engineering, Pusan National University, Busan 46241, Korea

Some CNN models can classify images with higher accuracy than humans for specific datasets, but there are problems that should be solved to operate in the real-world. The models should be tolerant to untrained data [13, 38]. However, a common assumption in deep learning is that the training dataset contains all types of data that exist in real-world operating environments [14, 15]. This assumption can be easily violated in the real-world, and if untrained data are input into the CNN, they are classified as trained with high confidence [16].

In real-world operating environments, the performance of CNNs can be degraded owing to data of untrained types, which limits its operability [17]. Misclassification, particularly in safety–critical systems based on artificial intelligence, can lead to catastrophic consequences [18]. For example, autonomous vehicles can misclassify untrained elements in environments such as deserts or countryside where lanes and traffic signals do not exist, countries with different traffic signal systems, and roads with newly added signs. This can have a detrimental effect on human life, property, etc. Thus, studies have been conducted to solve these problems [19–21].

Previous studies use information from data or models to identify the data of types that were not used to train DNNs. Data-based studies analyze the characteristics of data using the Euclidean distance [22] or extreme value theory [23]. Although data-based studies can identify data of untrained types at a lower cost than model-based studies, the accuracy is low. Model-based studies identify the data of new untrained types by adding prototypes [16, 24] or adding or modifying extra structures or modules to the model [13, 40]. Model-based studies increase the computational cost and require more resources owing to the increased number of parameters in the model, which can affect its performance.

In the field of neuroscience, studies have been conducted to characterize the representation of brain regions [25, 26, 39]. Those studies have identified brain regions that perceive stimuli by providing input signals through sensory organs and characterizing the information recognized by specific brain regions. The studies conducted representational similarity analysis between the input signals and the brain regions and showed that if the input signals are similar, the brain regions that perceive stimuli are similar. The structure of the DNN is inspired by the human brain and works similarly to the human brain system [27–29]. Therefore, it is expected that a relation exists between the type of data input to the DNN and the specific region of the DNN. This study defines concepts and methods and conducts experiments to answer the following research questions.

- RQ1 (Characteristic): Does a set of neurons that play a major role in each class exist? Can a set of neurons identify class characteristics?
- RQ2 (Similarity): Can the similarity between neuron sets be used as a criterion to identify data from trained and untrained classes?

Inspired by neuroscience research, this study identifies the class neuron cluster (CNC), a set of neurons that are activated based on the class of input data, and analyzes the relation between the class of input data and the CNC. Based on the analysis results, we confirm whether the activated neurons are similar if the characteristics of the classes are similar. Furthermore, we identify data types that the model has not trained on using the CNCs.

First, we measure the criteria for identifying the data of classes that the model has not trained using sets of neurons activated by the training and validation data. Next, based on these criteria, we identify the data from the untrained classes.

The method proposed in this study uses only activated neuron information without adding or modifying the model structure; this has the advantages of low-computational cost and no impact on the classification performance of the model.

The main contributions of this study are as follows:

- We propose a CNC, which is a set of neurons used to identify a specific class. We show that the CNC recognizes class characteristics by conducting a similarity analysis between CNCs for the CIFAR-10 and STL-10 datasets and ResNet models.
- We propose a new model-based method for identifying untrained class data using the CNC similarity.
- We conduct experiments on public datasets (MNIST, CIFAR-10, and STL-10) to demonstrate the feasibility and effectiveness of the untrained class data identification method.

Section 2 describes some related work on the identification of the data of classes that the model has not trained on. Section 3 details the definition of the CNC, a set of neurons that are activated based on the class of input data, the identification of the CNC, and the analysis of the similarities between CNCs. Section 4 describes how to identify untrained class data. Section 5 presents the datasets, models, and configurations used in the experiments for identifying untrained class data. Section 6 presents and analyzes the experimental results. Section 7 provides the conclusions and the scope for future work. All variables and acronyms used in this paper are listed in Appendix 1.

2 Related work

The purpose of this study is to identify untrained class data. Therefore, the purpose is different from that of studies that identify modified data or outliers of known classes based on the generative adversarial network (GAN) and adversarial attack methods. Model training speed, performance improvement, or pruning for weight reduction are not addressed in this study.

Studies on the identification of untrained class data can be classified into two categories: data-based and model-based studies. In data-based study, Mendes et al. [22] proposed an open-set nearest neighbor method to identify untrained class data based on the Euclidean distance, depending on whether the label of the closest training data in the input data matched the label of the second closest training data. The method identifies trained and untrained class data with high accuracy of approximately 80% or more in experiments on simple datasets; however, the accuracy on complex datasets, such as Caltech-256, is lower than approximately 50%.

Zhang and Patel [23] proposed a sparse representation-based open-set recognition method to identify untrained class data by analyzing the difference between the input data and the training data using the extreme value theory method. The method identifies trained and untrained class data with a high accuracy of approximately 90% or more in experiments on simple datasets; however, the accuracy on complex datasets, including Caltech-256, is approximately 68–80%.

Similar to [23], Yu et al. [43] proposed the open set fault diagnostic method, which identifies untrained class data by analyzing it with the extreme value theory method. On industrial datasets, the proposed method increased the identification accuracy by approximately 10%.

In model-based study, Yang et al. [16] proposed generalized convolutional prototype learning with prototype loss for learning CNN models by adding prototype parameters. This method increases the classification accuracy by reducing the distance between the intra-class data and increasing the distance between the inter-class data based on the Euclidean distance. In addition, it improves the robustness of the CNN by training it to classify the added new class by adding a prototype for a new class. The method increases the accuracy by approximately 0.2% in the experiment on the MNIST dataset and by approximately 0.5% in the experiment on the CIFAR-10 dataset.

Similar to Yang et al. [16], Wang et al. [24] increased the classification accuracy by reducing the distance between the intra-class data and increasing the distance between the inter-class data using the CNN-based prototype ensemble technique. They identified untrained class data by calculating the novelty of each class using the open-world nearest mean classifier. The method identifies untrained class data with an accuracy of approximately 80% in the experiment on the simple characteristic Fashion-MNIST dataset and approximately 60% in the experiment on the complex CINIC dataset. Prototype-based methods, such as those of Yang et al. [16] and Wang et al. [24], require the addition of prototypes for the untrained class as parameters to the model, and an increase in the number of parameters increases the computational cost and required resources.

Gao et al. [41] proposed Convolutional open-world multi-task image Stream classifier with Intrinsic Similarity Metrics (CSIM), which identifies untrained class data based on similarity metrics. The method identified untrained class data with approximately 70% and less than 50% accuracy in experiments on MNIST and CIFAR-10 datasets, respectively.

Prototype-based methods, such as those of Yang et al. [16] and Wang et al. [24], require the addition of prototypes for the untrained class as parameters to the model, and an increase in the number of parameters increases the computational cost and required resources. These methods, including Gao et al. [41], require regular updating of model parameters to enhance the model's untrained class data identification performance.

Zhou et al. [44] identify untrained class data using the learning to classify with incremental new class method based on entropy and probability. When data are input, if the combined score of entropy and probability is greater than the threshold, it is classified as an untrained class, and the accuracy is approximately 2% higher than the comparative methods.

Ma et al. [45] proposed a method for identifying untrained class data in a generative adversarial network when the discriminator's score is greater than a threshold. The proposed method's accuracy was approximately 5% higher than the comparative methods.

Yoshihashi et al. [13] proposed classification-reconstruction learning for open-set recognition, a method of probabilistic identification of untrained class data using deep hierarchical reconstruction nets, designed based on an openmax classifier modified with softmax. The method improves the F1-score by approximately 0.6 in the experiments on the modified ImageNet and LSUN datasets.

Dudi and Rajesh [40] proposed the shark smell-based whale optimization algorithm (SS-WOA), which identifies untrained class data based on the activation function values and classification costs of the CNN model. The method derives a threshold using the activation function value when data are input and identify the input data as untrained class data when the classification cost is less than the threshold. The method improves the classification accuracy by approximately 0.57–4% compared with other classification models in the experiment on plant-leaf datasets. The methods proposed in [13] and [40] increase the computational cost because separate modules (extra layer and algorithm) are added to the CNN model.

The untrained class data identification method proposed in this study differs from data-based methods in that it uses model information. In addition, it does not modify the model, such as a loss function or classifier, or add new modules to identify untrained class data; it uses only the activation information of the neurons in the trained model. Thus, because the method is model-independent, it has the advantages of low computational cost and no influence on the classification performance of the model, unlike previous model-based methods.

3 Class neuron cluster

Here, we define the CNC as a set of neurons that are activated by the input data of a particular type (class). Then, we identify the CNC for each class of input data and analyze the relation between the class of input data and the CNC.

3.1 Definition

Neurons recognize the characteristics of the input data and transmit the recognized information to the next layer [30]. The CNC is a set of neurons that recognize the characteristics of the data of a particular class, and these neurons are activated when the data of a particular class are input. Before defining the CNC, we define a function that determines whether a neuron is activated as in Definition 1.

Definition 1 Neuron activation function.

- A set of neurons: $N = \{n_1, n_2, \ldots, n_l\}$
- A set of classes: $S = \{c_1, c_2, \ldots, c_m\}$
- A set of data in classc: $D_c = \{d_1, d_2, \ldots, d_n\}$

For input data d, if the activation value of the neuron exceeds 0, neuron n is determined to be activated by data d. For activation function f, the function used to determine the activation of the neuron is expressed in Eq. (1).

$$\text{active}(n, d) = \begin{cases} 1, & \text{if } f(n, d) > 0 \\ 0, & \text{else} \end{cases} \tag{1}$$

The CNC is a set of neurons that are activated more than a certain rate by the data of a particular class. Therefore, the neuron activation ratio is defined as in Definition 2 based on the neuron activation function in Definition 1.

Definition 2 Neuron activation ratio.

The activation ratio of neuron n for dataset D_c of a particular class c is the ratio of data that activates the neuron among the total data of the dataset, and it is expressed in Eq. (2).

$$\text{active\_ratio}(n, D_c) = \frac{\Sigma_{d \in D_c} \text{active}(n, d)}{|D_c|} \tag{2}$$

When the data of a particular class are input, a CNC, a set of neurons that are activated more than a certain rate, is defined as Definition 3 based on Definitions 1 and 2.

Definition 3 Class neuron cluster.

The CNC_c of a particular class c is a set of neurons whose neuron activation ratio is greater than or equal to the threshold for dataset D_c and is expressed as Eq. (3).

$$\text{CNC}_c = \{n \in N | \text{active\_ratio}(n, D_c) \geq \text{threshold}\} \tag{3}$$

For example, as depicted in ?tic=?>Fig. 1, when the number of data of class c is 100 and the threshold is set at 0.5, neurons activated more than 50 times are identified as the CNC of class c.

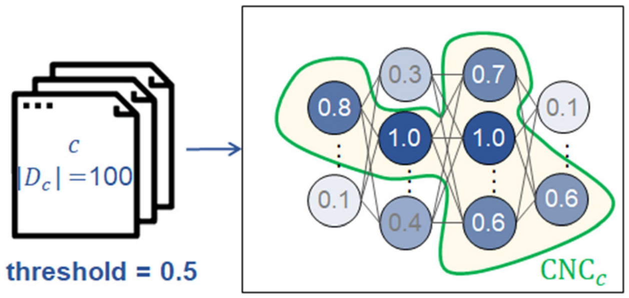


Fig. 1 Example of a CNC

3.2 Class neuron cluster similarity analysis

To verify that the previously defined CNC recognizes the characteristics of the input data, we identify CNCs for each class of input data and analyze the similarities between CNCs.

3.2.1 Class neuron cluster similarity analysis approach

A CNC is a set of neurons used to recognize the characteristics of data. Therefore, when the CNC is measured for each class, the CNCs of classes with similar characteristics are expected to have many activated neurons in common. Conversely, the CNCs of classes with only slightly similar characteristics are expected to have fewer commonly activated neurons. To confirm this, we analyze the similarity between CNCs. Figure 2 shows an overview of the CNC similarity analysis.

When the model that trains the training data composed of m classes ($S = \{c_1, c_2, \ldots, c_m\}$) D_S^{tr} is M_S, we identify the CNC for each class by inputting D_S^{tr} into M_S. Then, we measure the similarities between CNCs and analyze the similarity between CNCs by representing them as a similarity matrix. The method for measuring the similarity between CNCs is described in Definition 4.

Definition 4 Similarity measurement.

The similarity between two clusters, CNC_{c_i} and CNC_{c_j}, is measured as expressed in Eq. (4).

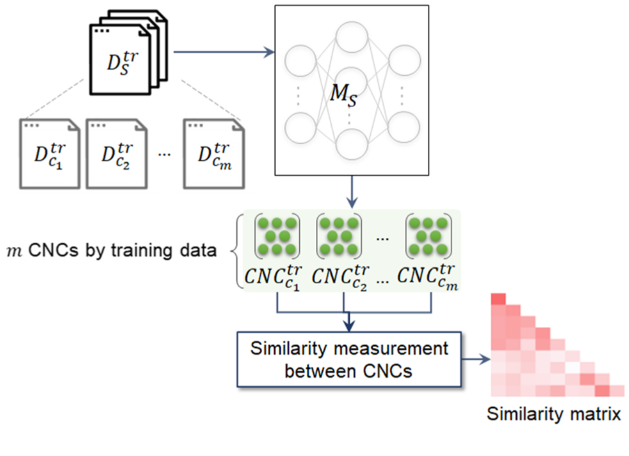


Fig. 2 Overview of the CNC similarity analysis

$$\mathrm{CNC}_{c_i} \oplus \mathrm{CNC}_{c_j} = \frac{\mathrm{CNC}_{c_i} \cap \mathrm{CNC}_{c_j}}{\mathrm{CNC}_{c_i} \cup \mathrm{CNC}_{c_j}} \quad (4)$$

When the similarity between two CNCs is the proportion of neurons that are commonly activated as described in Definition 4, we measure all the similarities between the identified CNCs and express them as a similarity matrix.

The CNC similarity analysis is conducted for the CIFAR-10 [31] and STL-10 [32] datasets and the ResNet model. CIFAR-10 and STL-10 are datasets comprising 10 classes (living-being: 6 and object: 4); CIFAR-10 has 5,000 training data, and STL-10 has 500 training data for each class.

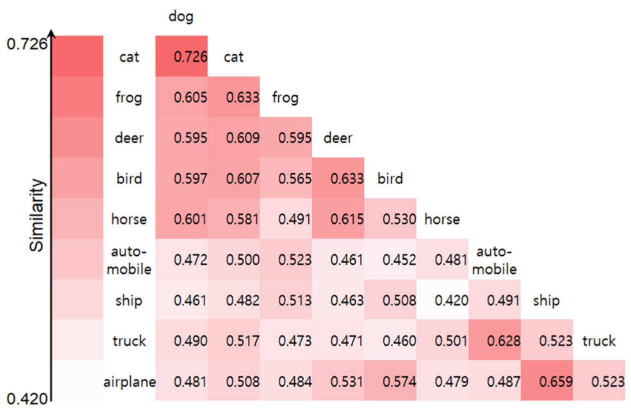
We analyze the similarity in the CIFAR-10 dataset for 24,576 neurons in the last six layers of the ResNet-20 model, and the STL-10 dataset for 368,640 neurons in the last 10 layers of the ResNet-32 model. The reason for targeting the latter layers is explained in Sect. 5.

Analyzing the CNC similarity identifies CNCs by inputting 500 training data for each class into the model, measures all similarities between the identified CNCs, and expresses them as a similarity matrix.

3.2.2 Class neuron cluster similarity analysis results

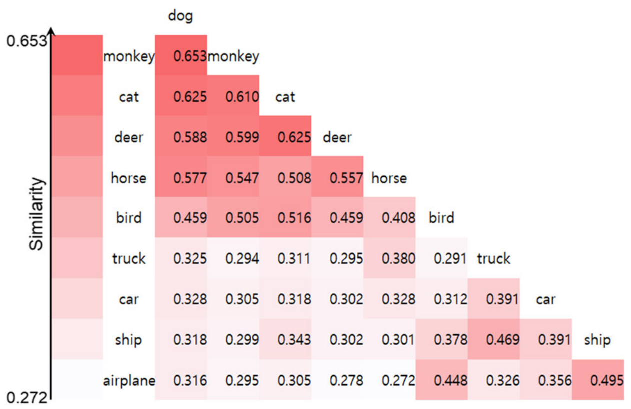
When the threshold of the neuron activation ratio is set at 0.5, the similarity matrixes representing the results of the CNC similarity analysis with 10 classes of CIFAR-10 and STL-10 are presented in Figs. 3 and 4, respectively.

The analysis results on the CIFAR-10 dataset show that the similarities between living-being–living-being pairs and between object–object pairs are higher than the similarities between living-being–object pairs. In the living-being–living-being pairs, the cat–dog pair has the highest similarity, and in the object–object pairs, the ship–airplane pair has the highest similarity. The pair with the lowest similarity is the horse–ship pair, which is a living-being–object pair.



	dog	cat	frog	deer	bird	horse	auto-mobile	ship	truck
cat	0.726								
frog	0.605	0.633							
deer	0.595	0.609	0.595						
bird	0.597	0.607	0.565	0.633					
horse	0.601	0.581	0.491	0.615	0.530				
auto-mobile	0.472	0.500	0.523	0.461	0.452	0.481			
ship	0.461	0.482	0.513	0.463	0.508	0.420	0.491		
truck	0.490	0.517	0.473	0.471	0.460	0.501	0.628	0.523	
airplane	0.481	0.508	0.484	0.531	0.574	0.479	0.487	0.659	0.523

Fig. 3 Similarity matrix between the CNCs of CIFAR-10



	dog	monkey	cat	deer	horse	bird	truck	car	ship
monkey	0.653								
cat	0.625	0.610							
deer	0.588	0.599	0.625						
horse	0.577	0.547	0.508	0.557					
bird	0.459	0.505	0.516	0.459	0.408				
truck	0.325	0.294	0.311	0.295	0.380	0.291			
car	0.328	0.305	0.318	0.302	0.328	0.312	0.391		
ship	0.318	0.299	0.343	0.302	0.301	0.378	0.469	0.391	
airplane	0.316	0.295	0.305	0.278	0.272	0.448	0.326	0.356	0.495

Fig. 4 Similarity matrix between the CNCs of STL-10

The analysis results on the STL-10 dataset, similar to those on the CIFAR-10 dataset, show that the similarities between living-being–living-being pairs and object–object pairs are higher than those between living-being–object pairs. In the living-being–living-being pairs, the dog–monkey pair has the highest similarity, and in the object–object pairs, the ship–airplane pair has the highest similarity. The pair with the lowest similarity is the horse–airplane pair, which is a living-being–object pair.

The similarity analysis results indicate that the characteristics of classes with high similarity between CNCs are more similar than those with low similarity, and the CNC is a set of neurons that recognizes the data characteristics of a particular class.

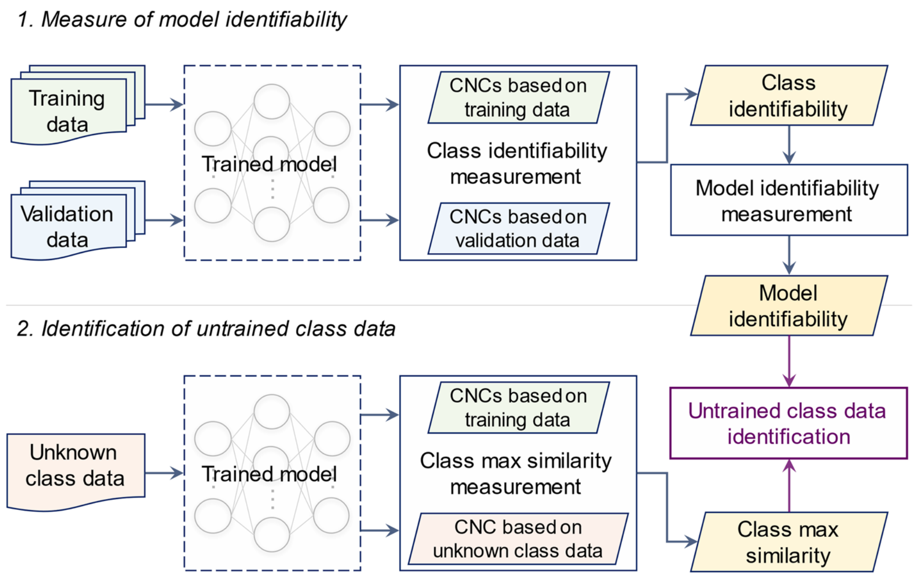
As described in Definition 3, CNC identification depends on the threshold. When the threshold is small, neurons activated by a small amount of data are included in the CNC, so that the CNC can be identified as coarse-grained. A coarse-grained CNC can flexibly identify data of various shapes; however, the data may be mistaken for a different class. Conversely, when the threshold is large, the CNC can be identified as fine-grained, which may not recognize data in various shapes and may result in poor flexibility. We conducted a similarity analysis based on the thresholds on the CIFAR-10 dataset and ResNet-20 model, and the analysis results are described in Appendix 2.

4 Untrained class data identification approach

The identification of untrained class data consists of a measure of model identifiability and identification of untrained class data. Figure 5 shows a flowchart of the untrained class data identification approach.

The first step measures the model identifiability based on the training and validation data. The second step measures the maximum class similarity based on the unknown

Fig. 5 Flowchart of the untrained class data identification approach


1. Measure of model identifiability
Training data
Validation data
Trained model
CNCs based on training data
Class identifiability measurement
CNCs based on validation data
Class identifiability
Model identifiability measurement
Model identifiability
2. Identification of untrained class data
Unknown class data
Trained model
CNCs based on training data
Class max similarity measurement
CNC based on unknown class data
Untrained class data identification
Class max similarity

class data and training data and identifies the untrained class data by comparing it with the model identifiability. Each step is described in detail in the following subsections.

4.1 Measure of model identifiability

Model identifiability is a criterion for determining whether arbitrary data inputs to the model are data of classes that the model is not trained on and is measured based on class identifiability. Figure 6 shows an overview of model identifiability measurements.

We denote the training data excluding the data of class c_h as $D^{\mathrm{tr}}_{\overline{c_h}}$, the validation data excluding the data of c_h as $D^{\mathrm{va}}_{\overline{c_x}}$, and the model that is trained on training data $D^{\mathrm{tr}}_{\overline{c_h}}$ as $M_{\overline{c_h}}$. First, we input training data $D^{\mathrm{tr}}_{\overline{c_h}}$ and validation data $D^{\mathrm{va}}_{\overline{c_x}}$ into model $M_{\overline{c_h}}$ and identify the CNC for the training data and validation data, respectively. The validation data are selected from the original training data and do not overlap with the training data used for CNC identification.

Next, we measure the similarity between the CNCs of the training and validation data for each class. The similarity is used to identify each class. Class identifiability is a criterion for determining whether arbitrary data inputs to the model are data of a class that the model is trained on at the class level. Class identifiability is described in Definition 5.

Definition 5 Class identifiability.

The identifiability of a particular class c_k is the similarity between the $\mathrm{CNC}^{\mathrm{tr}}_{c_k}$ identified by inputting the training data, and the $\mathrm{CNC}^{\mathrm{va}}_{c_k}$ identified by inputting the validation data, as expressed in Eq. (5).

$$\mathrm{identifiability}_{c_k} = \mathrm{CNC}^{\mathrm{tr}}_{c_k} \oplus \mathrm{CNC}^{\mathrm{va}}_{c_k} \tag{5}$$

Model identifiability is a criterion for determining whether the input data are the data of the trained class at

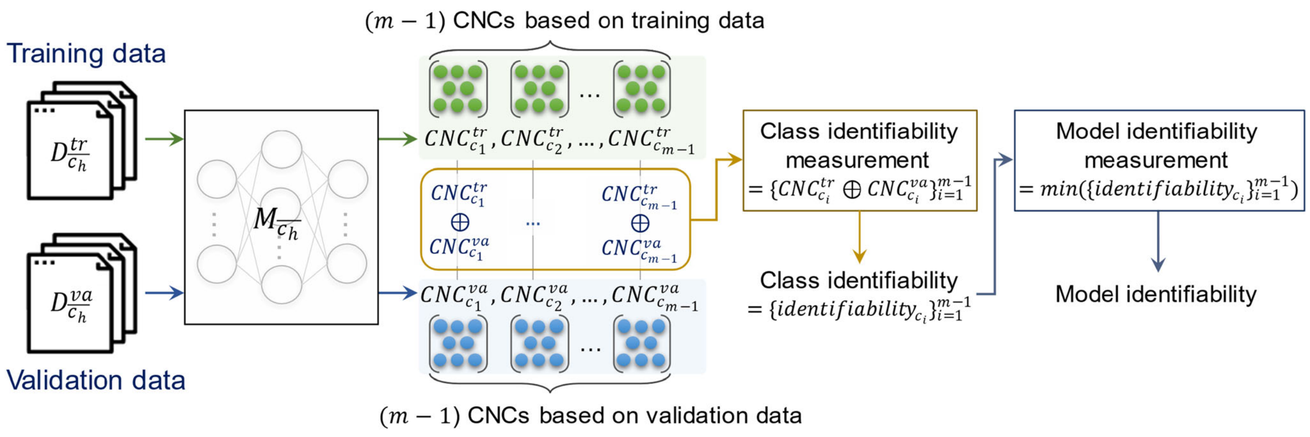

$(m-1)$ CNCs based on training data
Training data
$D^{tr}_{\overline{c_h}}$
$M_{\overline{c_h}}$
$CNC^{tr}_{c_1}, CNC^{tr}_{c_2}, \ldots, CNC^{tr}_{c_{m-1}}$
$CNC^{tr}_{c_1} \oplus CNC^{va}_{c_1}$ … $CNC^{tr}_{c_{m-1}} \oplus CNC^{va}_{c_{m-1}}$
Class identifiability measurement $= \{CNC^{tr}_{c_i} \oplus CNC^{va}_{c_i}\}_{i=1}^{m-1}$
Model identifiability measurement $= min(\{identifiability_{c_i}\}_{i=1}^{m-1})$
Class identifiability $= \{identifiability_{c_i}\}_{i=1}^{m-1}$
Model identifiability
$D^{va}_{\overline{c_h}}$
$CNC^{va}_{c_1}, CNC^{va}_{c_2}, \ldots, CNC^{va}_{c_{m-1}}$
Validation data
$(m-1)$ CNCs based on validation data

Fig. 6 Overview of the model identifiability measurement

the model level. As expressed in Definition 6, it is measured as the minimum value among the class identifiabilities.

Definition 6 Model identifiability.

The identifiability of model M_s trained on S composed of m classes is the minimum value among the identifiabilities of m classes, as expressed in Eq. (6).

$$\text{identifiability}_{M_S} = \min\left(\left\{\text{identifiability}_{c_i}\right\}_{i=1}^{m}\right) \quad (6)$$

4.2 Identification of untrained class data

Untrained class data are identified by measuring the similarity between the CNC identified based on the input data and the CNC identified based on the training data and comparing the similarity with the model identifiability. Figure 7 shows an overview of untrained class data identification.

Because a small number of data may not be sufficient in identifying neuron clusters, we augment unknown class data, $D_{c_h}^{\text{un}}$ with rotation, zoom, brightness, and shift [42] techniques. The augmented data are then used for $\text{CNC}_{c_h}^{\text{un}}$ identification. The unknown class data are selected from the test data.

We measure the class maximum similarity to determine whether the unknown class data are an untrained class. The class max similarity is measured by comparing the similarity between $\text{CNC}_{c_h}^{\text{un}}$ and CNCs based on the training data and is defined in Definition 7.

Definition 7 Class max similarity.

For the $\text{CNC}_{c_h}^{\text{un}}$ based on class c_h, the maximum similarity of c_h is the maximum of the similarities measured among the CNCs based on m training data and is expressed in Eq. (7).

$$\text{similarity}_{c_h}^{\max} = \max\left(\left\{\text{CNC}_{c_i}^{\text{tr}} \oplus \text{CNC}_{c_h}^{\text{un}}\right\}_{i=1}^{m}\right) \quad (7)$$

The untrained class data are identified by comparing the model identifiability measured in the previous step to the unknown class maximum similarity. The method for identifying the untrained class data is described in Definition 8.

Definition 8 Untrained class data identification.

If the unknown class max similarity, $\text{similarity}_{c_h}^{\max}$ based on class c_h, is less than the model identifiability, identifiability_M, c_h is identified as an untrained class, as expressed in Eq. (8).

$$c_h = \begin{cases} \text{untrained class}, & \text{if similarity}_{c_h}^{\max} < \text{identifiability}_M \\ \text{trained class}, & \text{else} \end{cases} \quad (8)$$

If the unknown class max similarity is less than the model identifiability, the characteristics of the data of the unknown class are considered to be dissimilar to the characteristics of the data of all classes that the model is trained on; therefore, the data of the unknown class are identified as untrained class data.

To verify the performance of the untrained class data identification method, we define class identification accuracy based on the accuracy used in the existing classification test [37]. The class identification accuracy is measured based on the identification results obtained by comparing the model identifiability and the class max similarity. It measures the degree of identification of not only untrained class data but also trained class data as trained class data. The class identification accuracy is described in Definition 9.

Definition 9 Class identification accuracy.

The class identification accuracy is the ratio of correctly classified trained and untrained class data to the total data, as expressed in Eq. 9.

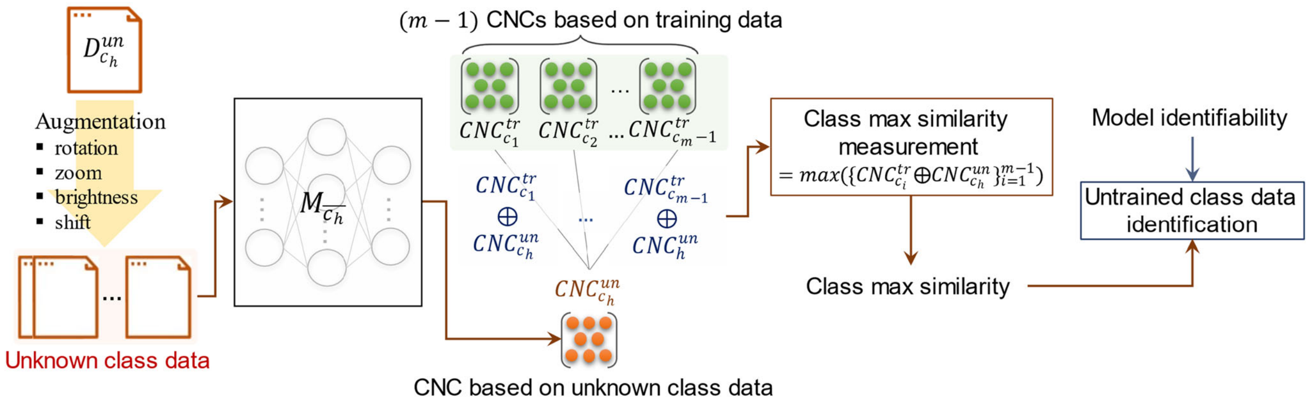

$D_{c_h}^{un}$
Augmentation
rotation
zoom
brightness
shift
Unknown class data
$M_{\overline{c_h}}$
$(m-1)$ CNCs based on training data
$CNC_{c_1}^{tr}$
$CNC_{c_2}^{tr}$
$CNC_{c_{m-1}}^{tr}$
$CNC_{c_1}^{tr} \oplus CNC_{c_h}^{un}$
$CNC_{c_{m-1}}^{tr} \oplus CNC_{h}^{un}$
$CNC_{c_h}^{un}$
CNC based on unknown class data
Class max similarity measurement
$= max(\{CNC_{c_i}^{tr} \oplus CNC_{c_h}^{un}\}_{i=1}^{m-1})$
Class max similarity
Model identifiability
Untrained class data identification

Fig. 7 Overview of untrained class data identification

$$\text{Class Identification Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{9}$$

- TP: Identify untrained class data as an untrained class
- TN: Identify trained class data as a trained class
- FP: Identify trained class data as an untrained class
- FN: Identify untrained class data as a trained class

To analyze the performance of the untrained class data identification method in detail, we measure the sensitivity, specificity, false-positive rate, and false-negative rate. Each performance metric is as follows.

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{10}$$

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{11}$$

$$\text{FNR(False negative rate)} = \frac{\text{FN}}{\text{TP} + \text{FN}} \tag{12}$$

$$\text{FPR(Flase positive rate)} = \frac{\text{FP}}{\text{TN} + \text{FP}} \tag{13}$$

5 Experimental design and settings

Here, we describe the experimental environments and methods used to measure model identifiability and identify the untrained class data.

5.1 Design to measure of model identifiability

Untrained class data are identified on the MNIST [33], CIFAR-10 [31], and STL-10 [32] datasets and ResNet [3], a CNN model. Table 1 presents the experimental datasets, models, and number of neurons to be measured.

MNIST is a 28 × 28 pixel image dataset comprising handwritten numbers from 0 to 9, and CIFAR-10 is a 32 × 32 pixel image dataset comprising six living-being classes (bird, cat, dog, deer, frog, and horse) and four object classes (airplane, automobile, ship, and truck). Similar to CIFAR-10, STL-10 is a 96 × 96 pixel image dataset comprising six living-being classes (bird, cat, dog, deer, horse, and monkey) and four object classes (airplane, car, ship, and truck).

Table 1 Experimental datasets and models

Dataset (# of class)	Model	# of neuron
MNIST (10)	Resnet-20	6 × (8, 8, 64) = 24,576
CIFAR-10 (10)	Resnet-20	6 × (8, 8, 64) = 24,576
STL-10 (10)	Resnet-32	10 × (24, 24, 64) = 368,640

The MNIST and CIFAR-10 datasets, which have a small image size, employ the ResNet-20 model, whereas the STL-10 dataset, which has a relatively large image size, employs the ResNet-32 model.

CNNs store more abstracted information as the layers deepen [34–36]. Neurons in the front layers identify lines and corners, whereas neurons in the rear layers identify objects based on the information from neurons in the front layers. Therefore, identifying the CNC, which is a set of neurons that recognize the data characteristics of a particular class, on the rear layers is advantageous. We conduct experiments on neurons in layers corresponding to the last 31% of the activation layers of the model. We experiment on 24,576 neurons in six layers with a size of 8 × 8 × 64 for the ResNet-20 model, and 368,640 neurons in 10 layers with a size of 24 × 24 × 64 for the ResNet-32 model.

The numbers of training and validation data used for CNC identification for each dataset are listed in Table 2. We use 500 training data for each class in the MNIST and CIFAR-10 datasets, and 300 training data for each class in the STL-10 dataset. The image size of STL-10 is three times larger than those of MNIST and CIFAR-10. The STL-10 dataset requires more time for experiments than the MINST and CIFAR-10 datasets; therefore, a smaller number of training data are used. On the MNIST and CIFAR-10 datasets, there are 500 validation data for each class, and on the STL-10 dataset, there are 200 validation data for each class.

Based on the CNCs identified using the training and validation data, we measure model identifiability. To identify untrained class data, in this study, we create a trained model $M_{\overline{c_x}}$, except for the data corresponding to one specific class c_x, from all classes in the dataset. Because each class in the dataset is excluded once, if m classes are present, a total of m models are created ($M_{\overline{c_1}}, M_{\overline{c_2}}, \ldots, M_{\overline{c_m}}$).

5.2 Design for identification of untrained class data

We identify untrained class data from unknown class data, which consists of nine trained class data and one untrained class data. The model should correctly identify the trained class data as trained class data and the untrained class data as untrained class data. The number of unknown class data is listed in Table 3.

The control parameters affecting this experiment are the number of data used for CNC identification and the activation rate threshold, which is the CNC identification criterion. Therefore, we conduct experiments by changing the

Table 2 Number of training/validation data used for CNC identification

Dataset	# of training data	# of validation data
MNIST	500/class × 9 class	100/class × 9 class
CIFAR-10	500/class × 9 class	100/class × 9 class
STL-10	300/class × 9 class	100/class × 9 class

Table 3 Number of unknown class data

Dataset	# of unknown class data	
	# of trained class data	# of untrained class data
MNIST	50/class × 9 class	50/class × 1 class
	40/class × 9 class	40/class × 1 class
CIFAR-10	30/class × 9 class	30/class × 1 class
	20/class × 9 class	20/class × 1 class
STL-10	10/class × 9 class	10/class × 1 class

values of the parameters. Each parameter is described as follows.

- The number of data used for CNC identification.
- The identification of a CNC, which is a set of neurons that recognize the characteristics of data, has a limitation with only one data. Therefore, we compare the results based on the number of data used for the CNC identification.
- The models used in the experiment train only nine out of 10 classes, such as the unknown class data configuration. We conducted experiments on 10 models by excluding each class once and repeated them five times with different numbers of unknown class data used for CNC identification, thereby performing 50 experiments for each dataset.
- Activation rate threshold.
- As described in Sect. 3, if the threshold is changed, the CNC is identified differently, and the result of untrained class data identification may be different. Therefore, we proceed with the experiment by changing the threshold used for the CNC identification to 0.3, 0.5, and 0.7. The experimental results mainly describe the results when the threshold is set at 0.5, and the results when the threshold is set at 0.3 and 0.7 are described in detail in Appendices.

5.3 Configurations

All experiments were conducted on a server with a Windows 10 Pro OS, AMD Ryzen 7 2700X CPU, 64.0 GB RAM, and an NVIDIA GeForce RTX 2080 Ti GPU. The MNIST and CIFAR-10 datasets are obtained from Keras v2.6.0, and the STL-10 dataset[1] and ResNet model[2] are available online.

6 Experimental results

Here, we describe the results of the measurement of model identifiability and identification of untrained class data based on model identifiability.

6.1 Results of measure of model identifiability

To measure the model identifiability, we identified CNCs by using the training and validation data of the MNIST, CIFAR-10, and STL-10 datasets. Figure 8 presents a box plot depicting the CNC size distribution identified by inputting the training data into 10 models ($M_{\overline{\mathrm{airplane}}}$, $M_{\overline{\mathrm{automobile}}}$, $M_{\overline{\mathrm{bird}}}$, $M_{\overline{\mathrm{cat}}}$, $M_{\overline{\mathrm{deer}}}$, $M_{\overline{\mathrm{dog}}}$, $M_{\overline{\mathrm{frog}}}$, $M_{\overline{\mathrm{horse}}}$, $M_{\overline{\mathrm{ship}}}$, $M_{\overline{\mathrm{truck}}}$) trained with nine classes, with the exception of one class from the training data of the CIFAR-10 dataset.

The CNC size distribution for each class is measured by using the nine models. For example, the CNC size distribution of the airplane is the CNC size of the airplane class measured from models ($M_{\overline{\mathrm{automobile}}}$, $M_{\overline{\mathrm{bird}}}$, $M_{\overline{\mathrm{cat}}}$, $M_{\overline{\mathrm{deer}}}$, $M_{\overline{\mathrm{dog}}}$, $M_{\overline{\mathrm{frog}}}$, $M_{\overline{\mathrm{horse}}}$, $M_{\overline{\mathrm{ship}}}$, $M_{\overline{\mathrm{truck}}}$) trained on the airplane data.

The CNCs are identified in sizes of approximately 3,700–4,800, accounting for approximately 15.1–19.5% of the total number of 24,576 neurons. The smallest CNC has a size of 3,785: the bird-class CNC ($\mathrm{CNC}^{\mathrm{tr}}_{\mathrm{bird}}$) of the model without the frog-class ($M_{\overline{\mathrm{frog}}}$), and the largest CNC has a size of 4,727: the truck-class CNC ($\mathrm{CNC}^{\mathrm{tr}}_{\mathrm{truck}}$) of the model without the automobile-class ($M_{\overline{\mathrm{automobile}}}$). Figure 9 displays a box plot depicting the CNC size distribution identified by inputting the validation data of the CIFAR-10 dataset.

The CNC size distribution based on the validation data is similar to that of the training data. The CNCs are identified in sizes of approximately 3,700–4,800. The smallest CNC and the largest CNC are the bird-class CNC ($\mathrm{CNC}^{\mathrm{tr}}_{\mathrm{bird}}$) of the model without the frog-class ($M_{\overline{\mathrm{frog}}}$), and the truck-class CNC ($CNC^{\mathrm{tr}}_{\mathrm{truck}}$) of the model without the automobile-class ($M_{\overline{\mathrm{automobile}}}$), same as the training data.

The CNC sizes of the living-being classes (bird, cat, deer, dog, frog, and horse) are found to be smaller than the CNC sizes of the object classes (airplane, automobile, ship, and truck). This is because the data from living-being

[1] https://cs.stanford.edu/~acoates/stl10.

[2] https://github.com/keras-team/keras-applications/tree/master/keras_applications.

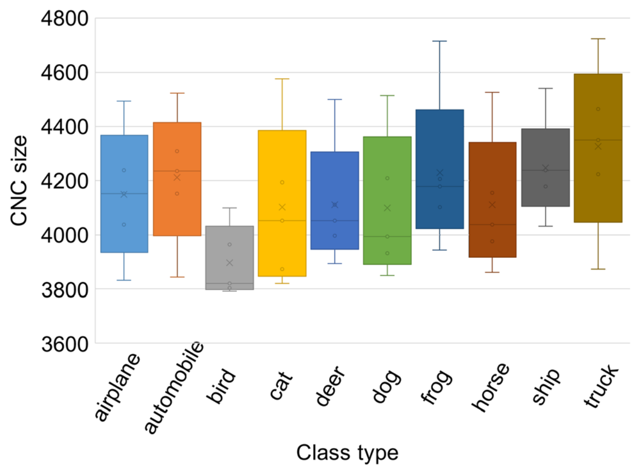


Fig. 8 Box plot for CNC size distribution based on the training data of CIFAR-10

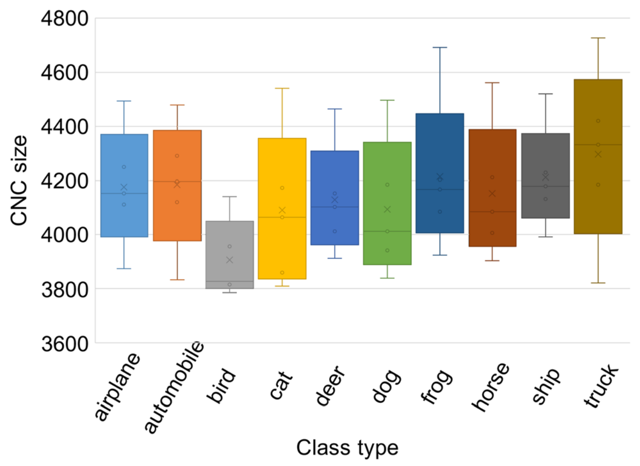


Fig. 9 Box plot for CNC size distribution based on the validation data of CIFAR-10

classes with complex characteristics activate various neurons. As a result, the number of neurons with a neuron activation rate greater than the threshold becomes small. For example, the CNC size of a bird with many curves and various shapes is identified to be smaller than that of a truck with many straight lines and simple shapes. The minimum, average, and maximum sizes of the CNCs for the training and validation data for each dataset are listed in Table 4.

Similar to the CIFAR-10 dataset, the MNIST and STL-10 datasets have similar size distributions of CNCs for the training and validation data. The CNC size of MNIST is approximately 1,600–2,500, which is approximately 6.5–10.2% of the total 24,576 neurons, and the CNC size of STL-10 is approximately 98,000–135,000, which is approximately 26.6–36.6% of the total 368,640 neurons.

The more complex the dataset, the more neurons are identified as clusters. The CNC of the STL-10 dataset with a large image size (96 $\times$ 96 pixel) is the largest, and the CNC of the MNIST dataset with a small image size (28 $\times$ 28 pixel) is the smallest.

Conversely, in the case of the MNIST dataset, more than 90% of neurons are not used as clusters. In the CIFAR-10 dataset, approximately 80% or more, and in the STL-10 dataset, approximately 64% or more neurons are not used as clusters.

When the threshold is set at 0.3, the size of the CNC is larger than when the threshold is set at 0.5. This is because the threshold is small; therefore, fewer activated neurons are included in the cluster. When the threshold is set at 0.7, the size of the CNC is the smallest. This is because only highly activated neurons are included in the cluster as the threshold is increased. The min/average/max CNC sizes per dataset for thresholds of 0.3 and 0.7 are described in Appendix 3.

The model identifiabilities measured using CNCs for the training and validation data for each dataset are listed in Table 5. Ten models are created for each dataset excluding one class. The average model identifiabilities of MNIST, CIFAR-10, and STL-10 are 0.656, 0.721, and 0.646, respectively.

The identifiability measured for each the model is used as a criterion for identifying the untrained class data. For example, for the MNIST dataset, the model without zero-class ($M_{\overline{\text{zero}}}$) identifies the unknown class data as the untrained class data if the maximum similarity between the training data and the unknown class data is less than 0.670, which is the model identifiability.

Model identifiability decreases as the threshold increases. This is because the larger the threshold, the smaller the size of the CNC; thus, the overlapping neurons between the CNCs by the training and validation data are small. When the thresholds are set at 0.3 and 0.7, the model identifiability for each dataset is described in Appendix 4.

6.2 Results of identification of untrained class data

To identify the untrained class data, we measured the class identification accuracy by inputting unknown class data into the model trained on part of the entire training data. The MNIST, CIFAR-10, and STL-10 datasets all consist of 10 classes, and we divided the unknown class data into nine trained class datasets and one untrained class dataset. Figure 10 displays the class identification accuracy of the dataset based on the number of unknown class data.

The relatively simple MNIST and CIFAR-10 datasets have higher class identification accuracy than the complex STL-10 dataset. The class identification accuracy in experiments with 10 data on the STL-10 dataset is 85%, and when more than 20 data are used, the accuracy is

Table 4 CNC min/average/max size of training and validation data

Dataset	Min	Average	Max
Training data			
MNIST	1609	2199	2538
CIFAR-10	3791	4138	4725
STL-10	98,969	116,824	132,509
Validation data			
MNIST	1588	2190	2538
CIFAR-10	3785	4133	4727
STL-10	99,691	117,893	135,440

Table 5 Model identifiability for each dataset

MNIST		CIFAR-10		STL-10	
Model	Ident	Model	Ident	Model	Ident
$M_{\overline{\text{zero}}}$	0.670	$M_{\overline{\text{airplane}}}$	0.730	$M_{\overline{\text{airplane}}}$	0.690
$M_{\overline{\text{one}}}$	0.650	$M_{\overline{\text{automobile}}}$	0.722	$M_{\overline{\text{bird}}}$	0.613
$M_{\overline{\text{two}}}$	0.641	$M_{\overline{\text{bird}}}$	0.725	$M_{\overline{\text{car}}}$	0.646
$M_{\overline{\text{three}}}$	0.655	$M_{\overline{\text{cat}}}$	0.714	$M_{\overline{\text{cat}}}$	0.638
$M_{\overline{\text{four}}}$	0.665	$M_{\overline{\text{deer}}}$	0.725	$M_{\overline{\text{deer}}}$	0.643
$M_{\overline{\text{five}}}$	0.660	$M_{\overline{\text{dog}}}$	0.720	$M_{\overline{\text{dog}}}$	0.634
$M_{\overline{\text{six}}}$	0.664	$M_{\overline{\text{frog}}}$	0.720	$M_{\overline{\text{horse}}}$	0.640
$M_{\overline{\text{seven}}}$	0.647	$M_{\overline{\text{horse}}}$	0.717	$M_{\overline{\text{monkey}}}$	0.646
$M_{\overline{\text{eight}}}$	0.649	$M_{\overline{\text{ship}}}$	0.720	$M_{\overline{\text{ship}}}$	0.659
$M_{\overline{nine}}$	0.660	$M_{\overline{\text{truck}}}$	0.721	$M_{\overline{\text{truck}}}$	0.646
Avg	0.656	Avg	0.721	Avg	0.646

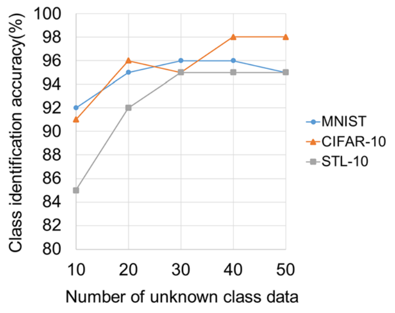


Fig. 10 Class identification accuracy for the datasets based on the number of unknown class data

greater than 90% on all datasets. The more data used for CNC identification, the more accurately the CNC can be identified by recognizing the class characteristics, so the class identification accuracy is high. Figure 11 displays the identification accuracy for each class based on the number of unknown class data in the MNIST dataset.

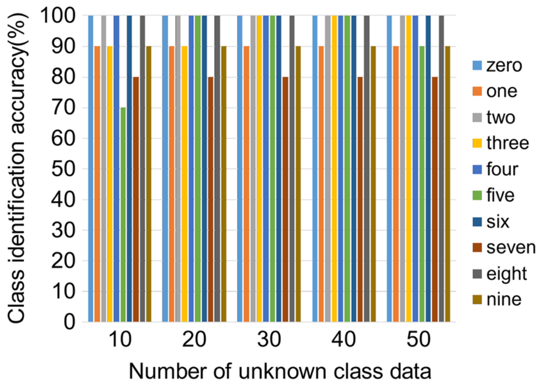


Fig. 11 Class identification accuracy on MNIST based on the number of unknown class data

In the experiment on the MNIST dataset, the ‘five’ class has the lowest identification accuracy when the number of unknown class data is ten, and the ‘one,’ ‘seven,’ and ‘nine’ classes have less than 90% accuracy in the data of 50 or less. We measured the sensitivity, specificity, FNR, and FPR to analyze the performance of the class data identification in detail. Table 6 lists the performance metrics on the MNIST dataset based on the number of unknown class data.

The sensitivity, which is the ratio of correctly classified untrained class data, is 100%; therefore, all untrained class data are correctly classified. Conversely, the specificity, which is the proportion of correctly classified trained class data, is greater than 91%, and some of the trained class data are incorrectly classified. Consequently, the FPR, which is the proportion of incorrectly classified trained class data, is less than 9%.

When the threshold used for CNC identification is set at 0.3, most of untrained class data are incorrectly classified as trained class data. Conversely, when the threshold is set at 0.7, trained class data are incorrectly classified as untrained class data. The results of measuring the performance metrics on the MNIST dataset for thresholds of 0.3 and 0.7 are described in Appendix 5. Figure 12 presents the identification accuracy for each class based on the number of unknown class data in the CIFAR-10 dataset.

Table 6 Performance metrics on MNIST based on the number of unknown class data

Average (%)	Number of unknown class data				
	10	20	30	40	50
Sensitivity	100	100	100	100	100
Specificity	91.11	94.44	95.56	95.56	94.44
FNR	0	0	0	0	0
FPR	8.89	5.56	4.44	4.44	5.56

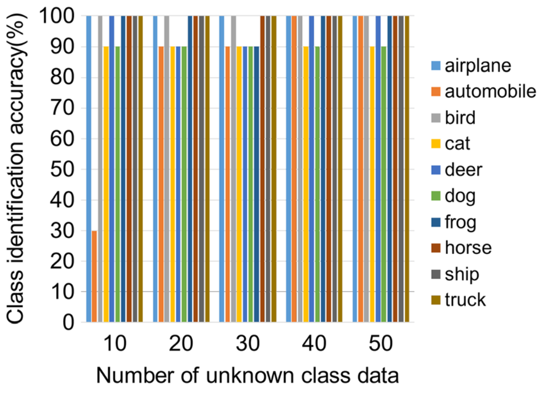


Fig. 12 Class identification accuracy on CIFAR-10 based on the number of unknown class data

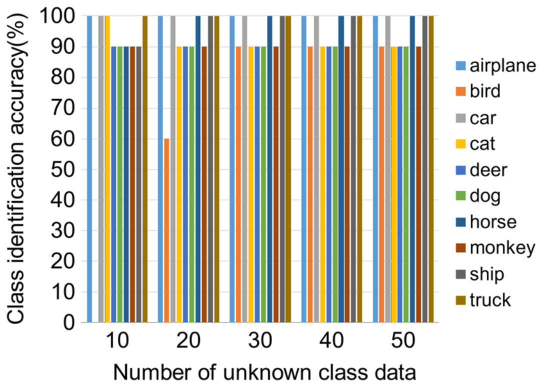


Fig. 13 Class identification accuracy on STL-10 based on the number of unknown class data

In the experiment on the CIFAR-10 dataset, the 'automobile' class has the lowest identification accuracy when the number of unknown class data is ten, and in all other cases, the accuracy has greater than 90%. Table 7 lists the other performance metrics on the CIFAR-10 dataset based on the number of unknown class data.

Because some untrained class data are incorrectly classified, the sensitivity is 60–80%, and the lowest is when the number of unknown class data is 30. In contrast, the majority of the trained class data are correctly classified, with the specificity greater than 92%. Overall, the higher the amount of data, the better the performance.

Similar to the MNIST dataset, in the experiment on the CIFAR-10 dataset, when the threshold used for CNC identification is set at 0.3, most of the untrained class data are incorrectly classified. In contrast, some trained class data are misclassified when the threshold is set at 0.7. The results of measuring the performance metrics on the CIFAR-10 dataset based on the threshold are described in Appendix 5. Figure 13 presents the identification accuracy for each class based on the number of unknown class data in the STL-10 dataset.

In the experiment on the STL-10 dataset, the identification accuracy of the 'bird' class is 0 when the number of unknown class data is ten, and 60% when the number of unknown class data is twenty. In all other cases, the accuracy has greater than 90%. Table 8 lists the other performance metrics on the STL-10 dataset based on the number of unknown class data.

Approximately half of the untrained class data are incorrectly classified, with a sensitivity of 50–60%. This is because untrained class data are identified as class data with similar characteristics. For example, a model trained on a dog but not a cat would classify a cat as a dog. The majority of the trained class data are correctly classified, with the specificity greater than 87%. The results of measuring the performance metrics on the STL-10 dataset based on the threshold are described in Appendix 5.

We compare the performance with CSIM [41] and CPE [24], which are CNN-based state-of-the-art untrained class data identification methods to examine the feasibility and effectiveness of CNC. Performance comparisons measure FNR and FPR on MNIST and CIFAR-10 datasets. Tables 9 and 10 list the performance metrics for each dataset of the methods.

The FNR of CNC is mostly lower than that of other methods, but the standard error is greater than that of other methods. This is because the untrained data identification performance varies for class to class. CNC identifies all untrained class data on the MNIST dataset but cat and dog classes on the CIFAR-10 dataset as different classes.

Table 7 Performance metrics on CIFAR-10 based on the number of unknown class data

Average (%)	Number of unknown class data				
	10	20	30	40	50
Sensitivity	80	70	60	80	80
Specificity	92.22	98.89	98.89	100	100
FNR	20	30	40	20	20
FPR	7. 78	1. 11	1. 11	0	0

Table 8 Performance metrics on STL-10 based on the number of unknown class data

Average (%)	Number of unknown class data				
	10	20	30	40	50
Sensitivity	60	50	50	50	50
Specificity	87.78	96.67	100	100	100
FNR	40	50	50	50	50
FPR	12.22	3.33	0	0	0

Table 9 Performance metric of methods on MNIST

Methods	MNIST	
(# of unknown class data)	FNR	FPR
CSIM	30.41 ± 3.15	0.10 ± 0.01
CPE	–	–
CNC (10)	0.00	8.89 ± 3.44
CNC (20)	0.00	5.56 ± 2.36
CNC (30)	0.00	4.44 ± 2.33
CNC (40)	0.00	4.44 ± 2.33
CNC (50)	0.00	5.56 ± 2.35

Table 10 Performance metric of methods on CIFAR-10

Methods (# of unknown class data)	CIFAR-10	
	FNR	FPR
CSIM	55.03 ± 0.53	4.60 ± 0.14
CPE	35.29 ± 0.22	3.93 ± 0.12
CNC (10)	20.00 ± 12.65	7.78 ± 7.38
CNC (20)	30.00 ± 14.49	1.11 ± 1.05
CNC (30)	40.00 ± 15.49	1.11 ± 1.05
CNC (40)	20.00 ± 12.65	0.00
CNC (50)	20.00 ± 12.65	0.00

On the CIFAR-10 dataset, the FPR of CNC is lower than other methods when the number of unknown class data is greater than 20. In other cases, however, it is higher than other methods.

CSIM and CPE identify the class of data when 300 data are input and store data predicted to be untrained class data in the buffer. Then, when the amount of data accumulated in the buffer reaches 1,000, the model trains on the data in the buffer. Therefore, the more data that are accumulated in the model, the better the model's performance.

CNC identifies untrained data using only the model's information without training the model. The performance of CNC depends on the amount of data used to identify neuron clusters. While CSIM and CPE require 300 data for class identification and 1,000 data for model training, CNC has comparable or better performance in identifying untrained class data with only 40 or more data.

Factors that greatly affect class identification accuracy are the number of data, data characteristics, and cluster identification threshold. If sufficient data are used to identify the CNC that recognizes the data characteristics, the class identification accuracy can be increased to 100%.

The class identification accuracy decreases as the data's characteristics become more complex. The STL-10 dataset, which has more complex characteristics than the MNIST and CIFAR-10 datasets, has lower class identification accuracy when the same amount of data is used.

A low threshold does not accurately identify untrained class data, resulting in a low sensitivity. A high threshold does not accurately identify the trained class data, resulting in a low specificity. Although the threshold can be set differently depending on the model or target dataset, based on the experimental results of this study, we recommend setting it to 0.5.

Classification of the untrained class data is beyond the model's capability; therefore, the model cannot classify them, and the performance of the model decreases with the number of untrained class data. For example, if 50 out of 100 data are untrained class data, the performance of the model cannot exceed 50%. Therefore, it is necessary to identify them so that they are not input into the model.

7 Conclusion

In this study, we proposed a method for identifying the data of classes that a DNN has not trained. We identified the CNC, a set of neurons that are activated based on the class of input data and confirmed that the CNC is a set of neurons that recognize the characteristics of data by conducting similarity analysis among the CNCs. In addition, the experiments on ResNet models and public datasets showed the feasibility and effectiveness of the untrained class data identification method based on CNC similarity. In future research, we plan to study methods that identify the class data that a model has trained, created by using GANs or mutated by adversarial attacks based on CNCs.

Appendix 1: All variables and acronyms

Variables	
N, n	A set of neurons,$n \in N$
S, c	A set of classes,$c \in S$
D_c, d	A set of data in class c,$d \in D_c$
D_S^{tr}	Training dataset
D_S^{va}	Validation dataset
D_S^{un}	Unknown class dataset
$D_{\overline{c}}$	A dataset except class $c = D_S - D_c$
M_S	A model trained dataset D_S
$M_{\overline{c}}$	A model trained dataset $D_{\overline{c}}$
$\mathrm{CNC}_c^{\mathrm{tr}}$	A class neuron cluster for class c of D_S^{tr}
$\mathrm{CNC}_c^{\mathrm{te}}$	A class neuron cluster for class c of D_S^{te}
$\mathrm{CNC}_c^{\mathrm{un}}$	A class neuron cluster for class c of D_S^{un}

Variables	
N, n	A set of neurons,$n \in N$
$\mathrm{identifiability}_c$	A class identifiability of class c
$\mathrm{identifiability}_M$	A model identifiability of model M
$\mathrm{similarity}_c^{\max}$	Class max similarity of class c
Acronyms	
CNC	Class neuron cluster

Appendix 2: CNC similarity analysis based on threshold

We analyzed the CNC similarity by changing the threshold on the CIFAR-10 dataset. Tables 11, 12, and 13 show the CNC similarity analysis results for thresholds of 0.3, 0.5, and 0.7

The top 10 similarities are occupied by living-being–living-being or object–object pairs with similar characteristics at all thresholds. However, in the bottom 10 similarities, only the living-being–object pairs exist when the threshold is set at 0.5, but the object–object pair (Airplane–Automobile) exists when thresholds are set at 0.3 and 0.7.

Table 11 Top 10 and bottom 10 similarities (threshold = 0.3)

	Top-10	Bottom-10
1	Cat–Dog	Automobile–Bird
2	Bird–Deer	Bird–Truck
3	Cat–Frog	Horse–Ship
4	Airplane–Ship	Automobile–Deer
5	Automobile–Truck	Deer–Truck
6	Deer–Horse	Dog–Ship
7	Bird–Dog	Automobile–Dog
8	Bird–Cat	Automobile–Horse
9	Dog–Horse	Airplane–Automobile
10	Cat–Deer	Deer–Ship

Table 12 Top 10 and bottom 10 similarities (threshold = 0.5)

	Top-10	Bottom-10
1	Cat–Dog	Horse–Ship
2	Airplane–Ship	Automobile–Bird
3	Cat–Frog	Bird–Truck
4	Bird–Deer	Dog–Ship
5	Automobile–Truck	Automobile–Deer
6	Deer–Horse	Deer–Ship
7	Cat–Deer	Deer–Truck
8	Bird–Cat	Automobile–Dog
9	Dog–Frog	Frog–Truck
10	Deer–Horse	Airplane–Horse

Table 13 Top 10 and bottom 10 similarities (threshold = 0.7)

	Top-10	Bottom-10
1	Cat–Dog	Bird–Truck
2	Airplane–Ship	Deer–Truck
3	Bird–Deer	Automobile–Bird
4	Automobile–Truck	Dog–Truck
5	Deer–Frog	Cat–Truck
6	Cat–Frog	Dog–Ship
7	Bird–Cat	Automobile–Deer
8	Deer–Horse	Cat–Ship
9	Bird–Dog	Airplane–Automobile
10	Dog–Frog	Automobile–Dog

Table 14 CNC min/average/max size of training and validation data (threshold = 0.3)

Dataset	Min	Average	Max
Training data			
MNIST	2,057	2,988	3,523
CIFAR-10	7,881	8,733	9,448
STL-10	196,343	255,934	286,411
Validation data			
MNIST	2,122	3,000	3,541
CIFAR-10	7,802	8,726	9,388
STL-10	200,789	257,569	287,261

Appendix 3: CNC min/average/max size based on threshold

We measured the min/average/max size of CNC for each dataset based on the threshold. Tables 14, 15, and 16 show the CNC sizes for thresholds of 0.3, 0.5, and 0.7.

When the threshold is set at 0.3, the CNC sizes are the largest. The CNC sizes of the CIFAR-10 and STL-10 datasets are more than twice as large as other thresholds. When the threshold is set at 0.7, the CNC sizes are the smallest. In particular, the CNC size of the STL-10 dataset decreases the most.

Table 15 CNC min/average/max size of training and validation data (threshold = 0.5)

Dataset	Min	Average	Max
Training data			
MNIST	1,609	2,199	2,538
CIFAR-10	3,791	4,138	4,725
STL-10	98,969	116,824	132,509
Validation data			
MNIST	1,588	2,190	2,538
CIFAR-10	3,785	4,133	4,727
STL-10	99,691	117,893	135,440

Table 16 CNC min/average/max size of training and validation data (threshold = 0.7)

Dataset	Min	Average	Max
Training data			
MNIST	1173	1529	1897
CIFAR-10	1615	1935	2207
STL-10	20,105	34,757	61,878
Validation data			
MNIST	1175	1518	1908
CIFAR-10	1608	1934	2176
STL-10	18,675	33,487	60,460

Appendix 4: Model identifiability based on threshold

We measured the model identifiability for each dataset based on the threshold. Tables 17, 18, and 19 show the results of model identifiability measurement for thresholds of 0.3, 0.5, and 0.7. As the threshold increases, the size of the average model identifiability decreases.

Table 17 Model identifiability for each dataset (threshold = 0.3)

MNIST		CIFAR-10		STL-10	
Model	Ident	Model	Ident	Model	Ident
$M_{\overline{\text{zero}}}$	0.679	$M_{\overline{\text{airplane}}}$	0.732	$M_{\overline{\text{airplane}}}$	0.731
$M_{\overline{\text{one}}}$	0.662	$M_{\overline{\text{automobile}}}$	0.731	$M_{\overline{\text{bird}}}$	0.736
$M_{\overline{\text{two}}}$	0.661	$M_{\overline{\text{bird}}}$	0.731	$M_{\overline{\text{car}}}$	0.740
$M_{\overline{\text{three}}}$	0.666	$M_{\overline{\text{cat}}}$	0.717	$M_{\overline{\text{cat}}}$	0.747
$M_{\overline{\text{four}}}$	0.679	$M_{\overline{\text{deer}}}$	0.722	$M_{\overline{\text{deer}}}$	0.747
$M_{\overline{\text{five}}}$	0.672	$M_{\overline{\text{dog}}}$	0.722	$M_{\overline{\text{dog}}}$	0.741
$M_{\overline{\text{six}}}$	0.677	$M_{\overline{\text{frog}}}$	0.726	$M_{\overline{\text{horse}}}$	0.744
$M_{\overline{\text{seven}}}$	0.656	$M_{\overline{\text{horse}}}$	0.720	$M_{\overline{\text{monkey}}}$	0.742
$M_{\overline{\text{eight}}}$	0.662	$M_{\overline{\text{ship}}}$	0.737	$M_{\overline{\text{ship}}}$	0.743
$M_{\overline{\text{nine}}}$	0.671	$M_{\overline{\text{truck}}}$	0.729	$M_{\overline{\text{truck}}}$	0.737
Avg	0.668	Avg	0.727	Avg	0.741

Table 18 Model identifiability for each dataset (threshold = 0.5)

MNIST		CIFAR-10		STL-10	
Model	Ident	Model	Ident	Model	Ident
$M_{\overline{\text{zero}}}$	0.670	$M_{\overline{\text{airplane}}}$	0.730	$M_{\overline{\text{airplane}}}$	0.690
$M_{\overline{\text{one}}}$	0.650	$M_{\overline{\text{automobile}}}$	0.722	$M_{\overline{\text{bird}}}$	0.613
$M_{\overline{\text{two}}}$	0.641	$M_{\overline{\text{bird}}}$	0.725	$M_{\overline{\text{car}}}$	0.646
$M_{\overline{\text{three}}}$	0.655	$M_{\overline{\text{cat}}}$	0.714	$M_{\overline{\text{cat}}}$	0.638
$M_{\overline{\text{four}}}$	0.665	$M_{\overline{\text{deer}}}$	0.725	$M_{\overline{\text{deer}}}$	0.643
$M_{\overline{\text{five}}}$	0.660	$M_{\overline{\text{dog}}}$	0.720	$M_{\overline{\text{dog}}}$	0.634
$M_{\overline{\text{six}}}$	0.664	$M_{\overline{\text{frog}}}$	0.720	$M_{\overline{\text{horse}}}$	0.640
$M_{\overline{\text{seven}}}$	0.647	$M_{\overline{\text{horse}}}$	0.717	$M_{\overline{\text{monkey}}}$	0.646
$M_{\overline{\text{eight}}}$	0.649	$M_{\overline{\text{ship}}}$	0.720	$M_{\overline{\text{ship}}}$	0.659
$M_{\overline{\text{nine}}}$	0.660	$M_{\overline{\text{truck}}}$	0.721	$M_{\overline{\text{truck}}}$	0.646
Avg	0.656	Avg	0.721	Avg	0.646

Table 19 Model identifiability for each dataset (threshold = 0.7)

MNIST		CIFAR-10		STL-10	
Model	Ident	Model	Ident	Model	Ident
$M_{\overline{\text{zero}}}$	0.666	$M_{\overline{\text{airplane}}}$	0.716	$M_{\overline{\text{airplane}}}$	0.679
$M_{\overline{\text{one}}}$	0.657	$M_{\overline{\text{automobile}}}$	0.735	$M_{\overline{\text{bird}}}$	0.632
$M_{\overline{\text{two}}}$	0.628	$M_{\overline{\text{bird}}}$	0.721	$M_{\overline{\text{car}}}$	0.625
$M_{\overline{\text{three}}}$	0.652	$M_{\overline{\text{cat}}}$	0.711	$M_{\overline{\text{cat}}}$	0.598
$M_{\overline{\text{four}}}$	0.661	$M_{\overline{\text{deer}}}$	0.718	$M_{\overline{\text{deer}}}$	0.596
$M_{\overline{\text{five}}}$	0.659	$M_{\overline{\text{dog}}}$	0.715	$M_{\overline{\text{dog}}}$	0.608
$M_{\overline{\text{six}}}$	0.666	$M_{\overline{\text{frog}}}$	0.708	$M_{\overline{\text{horse}}}$	0.598
$M_{\overline{\text{seven}}}$	0.640	$M_{\overline{\text{horse}}}$	0.716	$M_{\overline{\text{monkey}}}$	0.603
$M_{\overline{\text{eight}}}$	0.645	$M_{\overline{\text{ship}}}$	0.711	$M_{\overline{\text{ship}}}$	0.657
$M_{\overline{\text{nine}}}$	0.656	$M_{\overline{\text{truck}}}$	0.724	$M_{\overline{\text{truck}}}$	0.618
Avg	0.653	Avg	0.718	Avg	0.621

Appendix 5: Performance metric measurement results

MNIST

We measured various performance metrics based on the threshold on the MNIST dataset. Tables 20, 21, and 22 show the performance for thresholds of 0.3, 0.5, and 0.7.

When the threshold is set to 0.3, all trained class data are classified correctly, but some untrained class data are incorrectly classified. When the threshold is set to 0.7, untrained class data are correctly classified, while trained class data are incorrectly classified.

Table 20 Performance metrics on MNIST based on the number of unknown class data (threshold = 0.3)

Average (%)	Number of unknown class data				
	10	20	30	40	50
Accuracy	98	99	98	98	98
Sensitivity	80	90	80	80	80
Specificity	100	100	100	100	100
FNR	20	10	20	20	20
FPR	0	0	0	0	0

Table 21 Performance metrics on MNIST based on the number of unknown class data (threshold = 0.5)

Average (%)	Number of unknown class data				
	10	20	30	40	50
Accuracy	92	95	96	96	95
Sensitivity	100	100	100	100	100
Specificity	91.11	94.44	95.56	95.56	94.44
FNR	0	0	0	0	0
FPR	8.89	5.56	4.44	4.44	5.56

Table 22 Performance metrics on MNIST based on the number of unknown class data (threshold = 0.7)

Average (%)	Number of unknown class data				
	10	20	30	40	50
Accuracy	12	12	12	12	12
Sensitivity	100	100	100	100	100
Specificity	2.22	2.22	2.22	2.22	2.22
FNR	0	0	0	0	0
FPR	97.78	97.78	97.78	97.78	97.78

CIFAR-10

We measured various performance metrics based on the threshold on the CIFAR-10 dataset. Tables 23, 24, and 25 show the performance for thresholds of 0.3, 0.5, and 0.7.

Similar to the MNIST dataset, trained class data are correctly classified when the threshold is set to 0.3, while untrained class data are correctly classified when the

Table 23 Performance metrics on CIFAR-10 based on the number of unknown class data (threshold = 0.3)

Average (%)	Number of unknown class data				
	10	20	30	40	50
Accuracy	90	90	90	91	91
Sensitivity	0	0	0	10	10
Specificity	100	100	100	100	100
FNR	100	100	100	90	90
FPR	0	0	0	0	0

Table 24 Performance metrics on CIFAR-10 based on the number of unknown class data (threshold = 0.5)

Average (%)	Number of unknown class data				
	10	20	30	40	50
Accuracy	91	96	95	98	98
Sensitivity	80	70	60	80	80
Specificity	92.22	98.89	98.89	100	100
FNR	20	30	40	20	20
FPR	7.78	1.11	1.11	0	0

Table 25 Performance metrics on CIFAR-10 based on the number of unknown class data (threshold = 0.7)

Average (%)	Number of unknown class data				
	10	20	30	40	50
Accuracy	61	62	65	66	66
Sensitivity	100	100	100	100	100
Specificity	56.67	57.78	61.11	62.22	62.22
FNR	0	0	0	0	0
FPR	43.33	42.22	38.89	37.78	37.78

threshold is set to 0.7. Overall, the more data used to identify clusters, the better the model's performance.

STL-10

We measured various performance metrics based on the threshold on the STL-10 dataset. Tables 26, 27, and 28 show the performance for thresholds of 0.3, 0.5, and 0.7.

In STL-10, trained class data are classified correctly when the threshold is set to 0.3, and untrained class data are correctly classified when the threshold is set to 0.7, similarly to other datasets. Based on the experimental results, we recommend setting the threshold to 0.5.

Table 26 Performance metrics on STL-10 based on the number of unknown class data (threshold = 0.3)

Average (%)	Number of unknown class data				
	10	20	30	40	50
Accuracy	90	90	90	90	90
Sensitivity	0	0	0	0	0
Specificity	100	100	100	100	100
FNR	100	100	100	100	100
FPR	0	0	0	0	0

Table 27 Performance metrics on STL-10 based on the number of unknown class data (threshold = 0.5)

Average (%)	Number of unknown class data				
	10	20	30	40	50
Accuracy	85	92	95	95	95
Sensitivity	60	50	50	50	50
Specificity	87.78	96.67	100	100	100
FNR	40	50	50	50	50
FPR	12.22	3.33	0	0	0

Table 28 Performance metrics on STL-10 based on the number of unknown class data (threshold = 0.7)

Average (%)	Number of unknown class data				
	10	20	30	40	50
Accuracy	20	26	30	32	31
Sensitivity	100	100	100	100	100
Specificity	11.11	17.78	22.22	24.44	23.33
FNR	0	0	0	0	0
FPR	88.89	82.22	77.78	75.56	76.67

Acknowledgements This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. 2021R1F1A1060056).

Data availability The datasets generated during and/or analyzed during the current study are available from the corresponding author on reasonable request.

Declarations

Conflict of interest The authors declare that they have no conflict of interest with respect to the research, authorship and/or publication of this article.

References

1. Goodfellow I, Bengio Y, Courville A (2016) Deep Learning. MIT press, Cambridge
2. Bengio Y (2009) Learning deep architectures for AI. Now Publishing Inc, Norwell
3. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings IEEE Comput Soc Conf Comput Vis Pattern Recognit. p. 770–778. https://doi.org/10.1109/CVPR.2016.90
4. Krizhevsky A, Sutskever I, Hinton GE (2017) Imagenet classification with deep convolutional neural networks. Commun ACM 60:84–90. https://doi.org/10.1145/3065386
5. Schroff F, Kalenichenko D, Philbin J (2015) FaceNet: A unified embedding for face recognition and clustering. In: Proceedings IEEE Comput Soc Conf Comput Vis Pattern Recognit. p. 815–823. https://doi.org/10.1109/CVPR.2015.7298682

6. Xu Z, Yang Y, Hauptmann AG (2015) A discriminative CNN video representation for event detection. In: Proceedings IEEE Comput Soc Conf Comput Vis Pattern Recognit. p. 1798–1807. https://doi.org/10.1109/CVPR.2015.7298789
7. Gao H, Cheng B, Wang J et al (2018) Object classification using cnn-based fusion of vision and lidar in autonomous vehicle environment. IEEE Trans Ind Inform 14:4224–4230. https://doi.org/10.1109/TII.2018.2822828
8. Satılmış Y, Tufan F, Şara M et al (2019) CNN based traffic sign recognition for mini autonomous vehicles. Adv Intell Syst Comput 853:85–94. https://doi.org/10.1007/978-3-319-99996-8_8
9. Valiente R, Zaman M, Ozer S, Fallah YP (2019) Controlling steering angle for cooperative self-driving vehicles utilizing CNN and LSTM-based deep networks. In: IEEE Intell Veh Symp Proceedings. p. 2423–2428. https://doi.org/10.1109/IVS.2019.8814260
10. Hutchison C, Zizyte M, Lanigan PE, et al (2018) Robustness testing of autonomy software. In: proceedings IEEE/ACM 40th Int Conf Softw Eng Softw Eng Pract Track. p. 276–285. https://doi.org/10.1145/3183519.3183534
11. Calder M, Kolberg M, Magill EH, Reiff-Marganiec S (2003) Feature interaction: a critical review and considered forecast. Comput Netw 41:115–141. https://doi.org/10.1016/S1389-1286(02)00352-3
12. Modas A, Sanchez-Matilla R, Frossard P, Cavallaro A (2020) Toward Robust sensing for autonomous vehicles: an adversarial perspective. IEEE Signal Process Mag 37:14–23. https://doi.org/10.1109/MSP.2020.2985363
13. Yoshihashi R, Shao W, Kawakami R, et al (2019) Classification-reconstruction learning for open-set recognition. In:Proceedings IEEE/CVF Conf Comput Vis Pattern Recognit. P. 4016–4025. https://doi.org/10.1109/CVPR.2019.00414
14. Bendale A, Boult T (2015) Towards open world recognition. In: Proceedings IEEE Comput Soc Conf Comput Vis Pattern Recognit. p. 1893–1902. https://doi.org/10.1109/CVPR.2015.7298799
15. Torralba A, Efros AA (2011) Unbiased look at dataset bias. In: Proceedings IEEE Comput Soc Conf Comput Vis Pattern Recognit. p. 1521–1528. https://doi.org/10.1109/CVPR.2011.5995347
16. Yang HM, Zhang XY, Yin F, Liu CL (2018) Robust Classification with Convolutional Prototype Learning. In: Proceedings IEEE Comput Soc Conf Comput Vis Pattern Recognit. p. 3474–3482. https://doi.org/10.1109/CVPR.2018.00366
17. Sünderhauf N, Brock O, Scheirer W et al (2018) The limits and potentials of deep learning for robotics. Int J Rob Res 37:405–420. https://doi.org/10.1177/0278364918770733
18. Le MT, Diehl F, Brunner T, Knol A (2018) Uncertainty estimation for deep neural object detectors in safety-critical applications. In: IEEE Conf Intell Transp Syst Proceedings, ITSC 3873–3878. https://doi.org/10.1109/ITSC.2018.8569637
19. Mimlitz Z, Short A, Van Bossuyt DL (2016) Toward risk-informed operation of autonomous vehicles to increase resilience in unknown and dangerous environments. In: Proceedings ASME 2016 Int Des Eng Tech Conf. https://doi.org/10.1115/DETC2016-60002
20. Ganesha Perumal D, Srinivasan S, Subathra B et al (2016) MILP based autonomous vehicle path-planning controller for unknown environments with dynamic obstacles. Int J Heavy Veh Syst 23:350–369. https://doi.org/10.1504/IJHVS.2016.079271
21. Hu C, Qin Y, Cao H et al (2019) Lane keeping of autonomous vehicles based on differential steering with adaptive multivariable super-twisting control. Mech Syst Signal 125:330–346. https://doi.org/10.1016/j.ymssp.2018.09.011
22. Mendes Júnior PR, de Souza RM, de Werneck R, O et al (2017) Nearest neighbors distance ratio open-set classifier. Mach Learn 106:359–386. https://doi.org/10.1007/s10994-016-5610-8
23. Zhang H, Patel VM (2017) Sparse Representation-based open set recognition. IEEE Trans Pattern Anal Mach Intell 39:1690–1696. https://doi.org/10.1109/TPAMI.2016.2613924
24. Wang Z, Kong Z, Changra S, et al (2019) Robust high dimensional stream classification with novel class detection. In: Proceedings IEEE 35th Int Conf Data Eng. P. 1418–1429. https://doi.org/10.1109/ICDE.2019.00128
25. Kriegeskorte N, Mur M, Bandettini P (2008) Representational similarity analysis -connecting the branches of systems neuroscience. Front Syst Neurosci 2:1–28. https://doi.org/10.3389/neuro.06.004.2008
26. Kriegeskorte N, Mur M, Ruff DA et al (2008) Matching categorical object representations in inferior temporal cortex of man and monkey. Neuron 60:1126–1141. https://doi.org/10.1016/j.neuron.2008.10.043
27. McCulloch WS, Pitts W (1990) A logical calculus of the ideas immanent in nervous activity. Bull Math Biol 52:99–115. https://doi.org/10.1007/BF02459570
28. DiCarlo JJ, Zoccolan D, Rust NC (2012) How does the brain solve visual object recognition? Neuron 73:415–434. https://doi.org/10.1016/j.neuron.2012.01.010
29. Du C, Du C, Huang L, He H (2019) Reconstructing perceived images from human brain activities with bayesian deep multiview learning. IEEE Trans Neural Netw Learn Syst 30:2310–2323. https://doi.org/10.1109/TNNLS.2018.2882456
30. Du M, Liu N, Song Q, Hu X (2018) Towards explanation of DNN-based prediction with guided feature inversion. In: Proceedings 24th ACM SIGKDD Int Conf Knowl Discov Data Min. p. 1358–1367. https://doi.org/10.1145/3219819.3220099
31. Krizhevsky A (2009) Learning Multiple Layers of Features from Tiny Images. MS thesis Dept Comput Sci Toronto Univ, Canada
32. Coates A, Lee H, Ng AY (2011) An analysis of single-layer networks in unsupervised feature learning. J Mach Learn Res 15:215–223
33. LeCun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. Proc IEEE 86:2278–2323. https://doi.org/10.1109/5.726791
34. Kim J, Feldt R, Yoo S (2019) Guiding deep learning system testing using surprise adequacy. In: Proceedings IEEE/ACM 41st Int Conf Softw Eng. P. 1039–1049. https://doi.org/10.1109/ICSE.2019.00108
35. Eniser HF, Gerasimou S, Sen A (2019) deepfault: fault localization. In: Proceedings Int Conf Fundam Approaches to Softw Eng. p. 171–191. https://doi.org/10.1007/978-3-030-16722-6_10
36. Zeiler MD, Fergus R (2014) Visualizing and understanding convolutional networks. In: Proceedings Eur Conf Comput Vis. p. 818–833. https://doi.org/10.1007/978-3-319-10590-1_53
37. Metz CE (1978) Basic principles of ROC analysis. Semin Nucl Med 8:283–298. https://doi.org/10.1016/S0001-2998(78)80014-2
38. Wang D, Yu H, Wang D, Li G (2020) Face recognition system based on CNN. In: Proceedings 2020 Int Conf Comput Inf Big Data Appl CIBDA p. 2020. 470–473. https://doi.org/10.1109/CIBDA50819.2020.00111
39. Bobadilla-Suarez S, Ahlheim C, Mehrotra A et al (2020) Measures of neural similarity. Comput Brain Behav 3:369–383. https://doi.org/10.1007/s42113-019-00068-5
40. Dudi B, Rajesh V (2021) Optimized threshold-based convolutional neural network for plant leaf classification: a challenge towards untrained data. J Comb Optim. https://doi.org/10.1007/s10878-021-00770-w
41. Gao Y, Chandra S, Wang Z, Khan L (2018) Adaptive image stream classification via convolutional neural network with

intrinsic similarity metrics. 24TH ACM SIGKDD Conf Knowl Discov DATA Min. https://doi.org/10.48550/arXiv.1810.03966

42. Zhang K, Cao Z, Wu J (2020) Circular shift: an effective data augmentation method for convolutional neural network on image classification. In: Proceedings Int Conf Image Process ICIP 2020-Octob:1676–1680. https://doi.org/10.1109/ICIP40778.2020.9191303
43. Yu X, Zhao Z, Zhang X et al (2022) Deep-learning-based open set fault diagnosis by extreme value theory. IEEE Trans Ind Inform 18:185–196. https://doi.org/10.1109/TII.2021.3070324
44. Zhou DW, Yang Y, Zhan DC (2022) Learning to classify with incremental new class. IEEE Trans Neural Netw Learn Syst 33:2429–2443. https://doi.org/10.1109/TNNLS.2021.3104882
45. Ma X, Ji K, Zhang L et al (2022) An open set recognition method for sar targets based on multitask learning. IEEE Geosci Remote Sens Lett 19:1–5. https://doi.org/10.1109/LGRS.2021.3079418