IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Class-Adaptive Data Augmentation for Image Classification

JISU YOO[1] AND SEOKHO KANG[1]

[1]Department of Industrial Engineering, Sungkyunkwan University, Suwon 16419, South Korea

Corresponding author: Seokho Kang (s.kang@skku.edu).

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (Ministry of Science and ICT) under Grant NRF-2020R1C1C1003232.

ABSTRACT Data augmentation is a widely used regularization technique for improving the performance of convolutional neural networks (CNNs) in image classification tasks. To improve the effectiveness of data augmentation, it is important to find label-preserving transformations that fit the domain knowledge for a given dataset. In several real-world datasets, appropriate augmentation policies differ between classes, owing to their different characteristics. In this paper, we propose a class-adaptive data augmentation method that utilizes class-specific augmentation policies. First, we train the CNN without data augmentation. Subsequently, we derive a suitable augmentation policy for each class through an optimization procedure to maximize the degree of transformation while maintaining the label-preserving property of CNNs. Finally, we re-train the model using data augmentation based on derived class-specific augmentation policies. Through experiments using benchmark datasets with class-specific transformation constraints, we demonstrate that the proposed method achieves comparable or higher classification accuracy than the baseline methods using the same augmentation policy for all classes. Additionally, we confirm that the derived class-specific augmentation policies are consistent with the domain knowledge of each dataset.

INDEX TERMS image classification, data augmentation, class-adaptive data augmentation, hyperparameter optimization

I. INTRODUCTION

DATA augmentation plays an important role as a regularization technique for improving the generalization performance of convolutional neural networks (CNNs) in image classification tasks [1, 2]. Data augmentation synthetically increases the amount and diversity of a training dataset with randomly transformed images through label-preserving transformations. Owing to its effectiveness, data augmentation has been widely used to improve classification accuracy in various image classification tasks.

An appropriate augmentation policy for data augmentation can significantly improve the classification accuracy of CNNs. For a given dataset, it is necessary to determine an augmentation policy that specifies the types of transformations and their distributions to randomly transform images while preserving their class labels. However, if the labels of transformed images are different from those of original images, the classification accuracy of the CNN can be degraded by using the data augmentation.

A naive approach is a manual specification by human experts, based on the domain knowledge of the dataset [3–7]. If the domain knowledge is sufficient, an appropriate augmentation policy can be easily derived. However, domain knowledge may not be available in many real-world scenarios. Additionally, this approach generally requires a trial-and-error process and thus can be labor-intensive and time-consuming [8, 9].

Recently, attempts have been made to automatically search for an appropriate augmentation policy for a given training dataset in a data-driven manner [10–15]. Existing methods can be used to effectively apply data augmentation to improve the performance of a CNN in the absence of domain knowledge. These methods mainly focus on optimizing the augmentation policy to be dataset-specific, implying that every image in the dataset is randomly transformed in the same manner, regardless of the class label.

Our research motivation stems from the fact that appropriate augmentation policies can differ between classes. For example, in the digit images of MNIST and SVHN datasets, random horizontal and vertical flips preserve the class labels

of images for classes 0, 1, and 8, but not for the other classes. Random rotation over a wide range preserves the class labels only for class 0. When different classes correspond to different types of label-preserving transformations, consideration of class-specific characteristics can lead to a substantial improvement in data augmentation.

In this study, we propose a class-adaptive data augmentation method to understand the class-specific characteristics of a dataset without domain knowledge. To build a CNN, the proposed method consists of three stages: (1) CNN training without data augmentation; (2) optimization of class-specific augmentation policies; (3) CNN re-training with class-adaptive data augmentation. In the first stage, the CNN is trained using the original training dataset without data augmentation. In the second stage, a suitable augmentation policy for each class is derived through an optimization procedure to maximize the degree of transformation while maintaining the label-preserving property of the CNN. In the third stage, the CNN is re-trained using data augmentation based on class-specific augmentation policies. To examine whether the proposed method derives suitable class-specific augmentation policies, we conducted experiments using three benchmark datasets constituting classes with different label-preserving transformation characteristics.

The remainder of this paper is organized as follows. In section II, related studies are reviewed. In section III, the proposed method is introduced. section IV describes the experimental settings and results. In section V, the conclusions and future work are presented.

II. RELATED WORK

A. TRANSFORMATION OPERATIONS FOR DATA AUGMENTATION

In image classification tasks, various transformation operations are readily available for data augmentation. Traditional transformation operations slightly vary an existing image using geometric and photometric transformations [16]. Geometric transformation operations change the image geometry by changing the pixel positions [8, 17, 18]. Examples include rotation, shifting, zooming, shearing, and flipping. Photometric transformation operations change pixel values in the color channels of an image [2, 9, 19]. Examples include brightness, contrast, and color inversion. Recently, transformation operations that synthesize a new image by combining two existing images, such as Mixup [20], CutMix [21], and CutPaste [22] have been proposed.

In this study, geometric and photometric transformation operations are used to formulate class-specific augmentation policies. These operations are suitable for capturing and easy to interpret class-specific characteristics of a dataset.

B. OPTIMIZATION OF DATA AUGMENTATION POLICY

A data augmentation policy comprises transformations and their distributions. The distribution determines the degree of transformation of the image. For example, the rotation operation rotates an image by an angle randomly sampled from a predefined distribution, such as Uniform ($[-30°, 30°]$). The horizontal flip determines whether to flip an image depending on a distribution such as Bernoulli(0.5). To improve the data augmentation effectiveness, it is important to appropriately set the hyperparameters of the distributions to ensure that the augmentation policy does not change the original class labels of the images. On the one hand, if the distribution is set too narrow, images are not well diversified, thus reducing the data augmentation effect. On the other hand, if the distribution is too broad, there is a risk in that the class label of an image will not be preserved after transformation, which can negatively affect the classification performance.

While manually designing an augmentation policy for a dataset based on domain knowledge is challenging and often infeasible, recent studies have attempted to automatically determine an appropriate augmentation policy. Existing methods define the search space of an augmentation policy and optimize the hyperparameters of transformation operations in the policy in a data-driven manner. The methods for defining a search space can be categorized into dataset-level and instance-level approaches, based on the manner in which the search space is defined.

The dataset-level approach optimizes the augmentation policy to make it suitable for a given dataset. Cubuk et al. [10] used reinforcement learning based on proximal policy optimization to search for the best augmentation policy for a dataset, in which each candidate augmentation policy is directly evaluated by training the CNN. Ho et al. [11] leveraged population-based training to generate a non-stationary augmentation policy schedule that defines the best policy for each training epoch, instead of a fixed augmentation policy. Lim et al. [12] optimized the augmentation policy via efficient density matching based on Bayesian optimization, which does not require repeated CNN training for evaluating augmentation polices and thereby significantly reduced computational cost. Zhang et al. [13] formulated an augmentation policy search as an adversarial learning problem, wherein a policy network generates an augmentation policy to maximize the training loss of the CNN, while the CNN learns from augmented training data to improve generalization. Hataya et al. [14] and Li et al. [15] used a differentiable augmentation policy to simultaneously optimize the augmentation policy and train the CNN. These methods apply the same augmentation policy to all instances without considering the differences in the characteristics between classes.

The instance-level approach builds a policy network that predicts an image-specific augmentation policy suitable for each input image. Zhou et al. [23] used a policy network to assign different weights to transformed images in CNN training. Cheung and Yeung [24] used a policy network that adaptively generated the hyperparameters for the transformation operations of each image. For image-dependent data augmentation, these methods commonly require a policy network to be maintained, which increases the computational cost.

The proposed method aims to improve the effectiveness

of data augmentation for CNN training by leveraging class-specific augmentation policies. Compared with the dataset-level approach, optimizing the hyperparameters of individual polices to be class-specific contributes to further diversifying transformations while preserving class labels. Compared to the instance-level approach, the proposed method obtains class-specific augmentation policies using a single CNN without requiring any additional network to be trained.

III. METHODOLOGY

The problem situation addressed in this study is as follows. For a C-class image classification task, we suppose that the training set $\mathcal{D} = \{(\mathbf{X}_i, y_i)\}_{i=1}^{N}$ and validation set $\mathcal{D}' = \{(\mathbf{X}_i, y_i)\}_{i=N+1}^{N+N'}$ are given, where $\mathbf{X}_i$ represents the i-th image and $y_i \in \{1, \ldots, C\}$ represents the corresponding class label. In the training phase, we build a CNN f using the training and validation sets $\mathcal{D}$ and $\mathcal{D}'$. The input and output of the CNN f are an image $\mathbf{X}$ and its softmax response vector $f(\mathbf{X}) \in [0, 1]^C$, respectively, for predicting the class label y.

Data augmentation is applicable during the training to improve the classification accuracy of the CNN f. The conventional approach uses an augmentation policy $\mathcal{A}(\boldsymbol{\phi})$ comprising M transformation operations, where $\boldsymbol{\phi} = (\phi_1, \phi_2, \ldots, \phi_M)$ is the set of hyperparameters for the transformation operations involved in the policy. For the m-th operation, the hyperparameter ϕ_m specifies the distribution from which the degree of transformation can be sampled. At each training epoch, each training image $\mathbf{X}$ is randomly transformed using a transformation τ sampled from the augmentation policy $\mathcal{A}(\boldsymbol{\phi})$. Each transformed image $\tilde{\mathbf{X}} = \mathbb{E}_{\tau \sim \mathcal{A}(\boldsymbol{\phi})}[\tau(\mathbf{X})]$ is used as the alternative input of the CNN f.

This study aims to make data augmentation further diversify images with preserving their class labels during the training by reflecting class-specific characteristics in the dataset. Unlike the conventional approach, which uses the same augmentation policy regardless of the class, the proposed method introduces class-specific augmentation policies $\mathcal{A}(\boldsymbol{\phi}^1), \ldots, \mathcal{A}(\boldsymbol{\phi}^C)$ such that each image is randomly transformed in a different manner depending on its class label. For this, we derived suitable hyperparameters for individual classes, namely $\boldsymbol{\phi}^1, \ldots, \boldsymbol{\phi}^C$, by performing an optimization procedure using a CNN trained without data augmentation.

Figure 1 shows a schematic diagram of the proposed method consisting of three stages: (1) CNN training without data augmentation, (2) optimization of class-specific augmentation policies, and (3) CNN re-training with class-adaptive data augmentation. Algorithm 1 presents the pseudocode of the proposed method. In the pseudocode, the first, second, and third stages correspond to lines 2, lines 3–10, and 11, respectively. The following subsections describe each stage.

Algorithm 1 Class-Adaptive Data Augmentation

Require: training set $\mathcal{D} = \{(\mathbf{X}_i, y_i)\}_{i=1}^{N}$, validation set $\mathcal{D}' = \{(\mathbf{X}_i, y_i)\}_{i=N+1}^{N+N'}$
Ensure: CNN f
1: **procedure**
2: $f \leftarrow$ TRAIN$(\mathcal{D}, \mathcal{D}', \{\}, 0)$
3: **for** c = 1 to C **do**
4: $\mathcal{D}'_c = \{(\mathbf{X}_i, y_i) \in \mathcal{D}' | y_i = c\}$
5: **for** m = 1 to M **do**
6: $\phi_m^c \leftarrow \arg\max_{\phi_m^c} \mathcal{J}_m(\phi_m^c; f, \mathcal{D}'_c)$ (in Equation 1)
7: **end for**
8: $\boldsymbol{\phi}^c \leftarrow (\phi_1^c, \phi_2^c, \ldots, \phi_M^c)$
9: **end for**
10: $\Phi \leftarrow \{\boldsymbol{\phi}^1, \boldsymbol{\phi}^2, \ldots, \boldsymbol{\phi}^C\}$
11: $f \leftarrow$ TRAIN$(\mathcal{D}, \mathcal{D}', \Phi, \gamma)$
12: **end procedure**

13: **function** TRAIN$(\mathcal{D}, \mathcal{D}', \Phi, \gamma)$
14: $f \leftarrow$ initialize a CNN
15: **while** not termination condition on $\mathcal{D}'$ **do**
16: $\tilde{\mathcal{D}} \leftarrow \{(\tilde{\mathbf{X}}_i, y_i) | (\mathbf{X}_i, y_i) \in \mathcal{D}, \tilde{\mathbf{X}}_i = \mathbb{E}_{\tau \sim \mathcal{A}(\boldsymbol{\phi}^{y_i})}[\tau(\mathbf{X}_i)]\}$
17: $f \leftarrow$ update f for one epoch using $\tilde{\mathcal{D}}$
18: $\Phi \leftarrow \{\gamma \cdot \boldsymbol{\phi}^c | \boldsymbol{\phi}^c \in \Phi\}$
19: **end while**
20: **return** f
21: **end function**

A. TRAINING WITHOUT DATA AUGMENTATION

In this stage, we train a CNN f using the training set $\mathcal{D}$ without applying data augmentation, in which the validation set $\mathcal{D}'$ is used to monitor the validation performance. The training is terminated upon meeting a predefined termination condition on the validation set $\mathcal{D}'$. The training procedure is presented in Train function of Algorithm 1.

The trained CNN f is used in the optimization of class-specific augmentation policies. As the CNN f learns the characteristics of original images, we determine the degrees of transformations for individual transformation operations such that the augmentation policies do not change the class labels of images after they are transformed.

B. OPTIMIZATION OF CLASS-SPECIFIC AUGMENTATION POLICIES

This stage optimizes the hyperparameters for class-specific augmentation policies. For the augmentation policy of the c-th class, $\mathcal{A}(\boldsymbol{\phi}^c)$, M hyperparameters must be optimized: $\boldsymbol{\phi}^c = (\phi_1^c, \ldots, \phi_M^c)$. Each hyperparameter ϕ_m^c corresponds to the degree of transformation for the m-th transformation operation involved in the policy. In this study, we selected 10 transformation operations typically used in image augmentation. These operations are relatively simple and easy to interpret. Table 1 lists the transformation operations and their hyperparameters.

The objective here is to maximize the degree of transformations for the augmentation policies of individual classes while maintaining the label-preserving property. For the m-th transformation operation for the c-th class, the hyperparam-

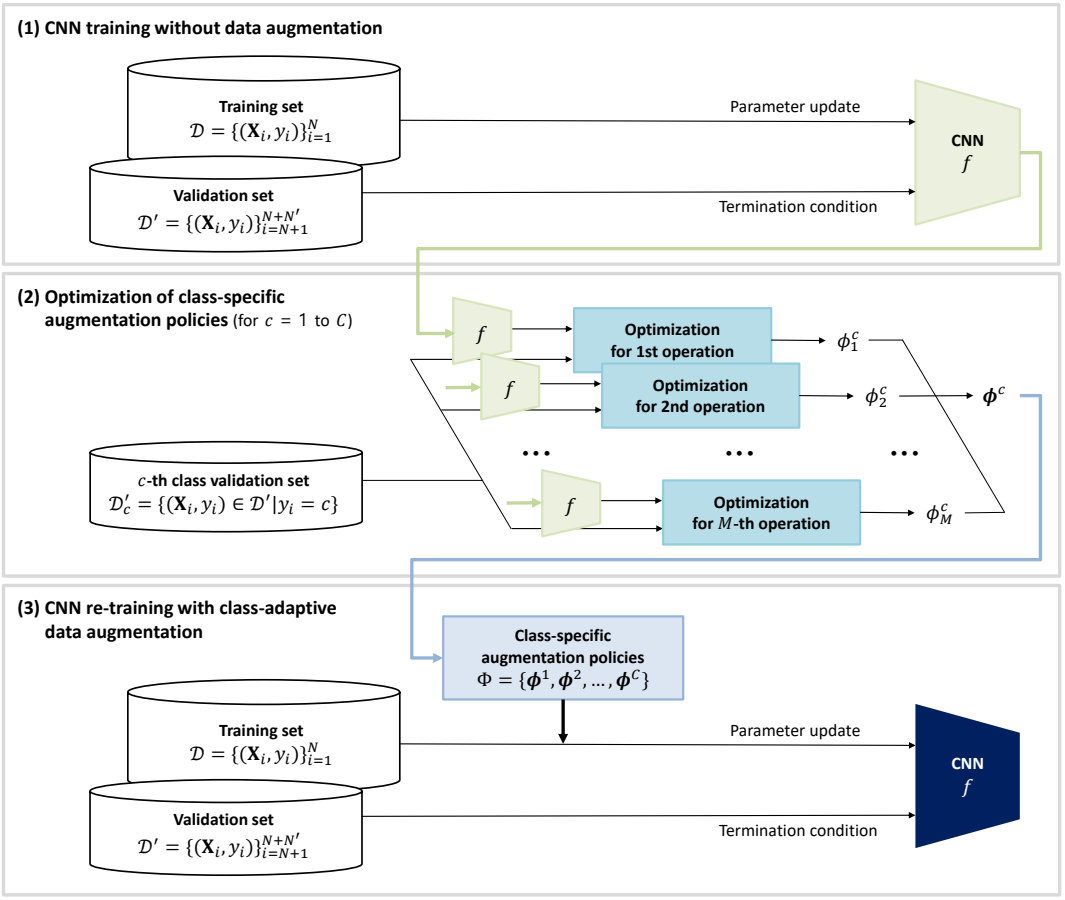


FIGURE 1: Framework of class-adaptive data augmentation

TABLE 1: Transformation operations and their hyperparameters in augmentation policy

Transformation Type	Hyperparameter	Search Space	Unit	Distribution
Rotation	ϕ_1	$[0, 90]$	Degree(°)	Uniform($[-\phi_1, \phi_1]$)
Shear	ϕ_2	$[0, 45]$	Degree(°)	Uniform($[-\phi_2, \phi_2]$)
Horizontal Shift	ϕ_3	$[0, 50]$	Percent(%)	Uniform($[-\phi_3, \phi_3]$)
Vertical Shift	ϕ_4	$[0, 50]$	Percent(%)	Uniform($[-\phi_4, \phi_4]$)
Zoom	ϕ_5	$[0, 50]$	Percent(%)	Uniform($[-\phi_5, \phi_5]$)
Brightness	ϕ_6	$[0, 50]$	Percent(%)	Uniform($[-\phi_6, \phi_6]$)
Contrast	ϕ_7	$[0, 50]$	Percent(%)	Uniform($[-\phi_7, \phi_7]$)
Color Invert	ϕ_8	0 or 0.5	Probability	Bernoulli(ϕ_8)
Horizontal Flip	ϕ_9	0 or 0.5	Probability	Bernoulli(ϕ_9)
Vertical Flip	ϕ_{10}	0 or 0.5	Probability	Bernoulli(ϕ_{10})

eter ϕ_m^c is optimized by solving the optimization problem below:

$$\underset{\phi_m^c}{\text{maximize}}\ \mathcal{J}_m(\phi_m^c; f, \mathcal{D}_c^{'}) = \min(\epsilon - \Delta_m(\phi_m^c; f, \mathcal{D}_c^{'}), 0) + \lambda \cdot \frac{\phi_m^c}{u_m}$$
$$\text{subject to} \begin{cases} 0 \le \phi_m^c \le u_m, & \text{if } m \in \{1,2,3,4,5,6,7\} \\ \phi_m^c = 0 \text{ or } u_m, & \text{if } m \in \{8,9,10\} \end{cases} \tag{1}$$

where $\mathcal{D}_c'$ is the subset of the validation set $\mathcal{D}'$ for the c-th class (i.e., $\mathcal{D}_c' = \{(\mathbf{X}_i, y_i) \in \mathcal{D}' | y_i = c\}$) and u_m is the upper bound in the search space of ϕ_m^c specified in Table 1.

In the objective function $\mathcal{J}_m$, the first term provides a penalty with a tolerance of ϵ if the softmax response output from the CNN f changes significantly when the input image is transformed by a degree of ϕ_m^c. The change in the softmax response is quantified using the function Δ_m as $\Delta_m(\phi_m^c; f, \mathcal{D}_c^{'})$. If $\Delta_m(\phi_m^c) < \epsilon$, the first term does not affect the objective function $\mathcal{J}_m$. The second term provides a preference of increasing the value of the hyperparameter ϕ_m^c. Because the search space of ϕ_m^c differs between transformation operations, we normalize ϕ_m^c by its upper bound u_m. The hyperparameter λ balances between the two terms.

If $m \in \{1, 2, 3, 4, 5, 6, 7\}$, the search space for the hyperparameter ϕ_m^c has a continuous range. The function Δ_m is set as follows:

$$\Delta_m(\phi_m^c; f, \mathcal{D}_c^{'}) = \frac{1}{2 \cdot |\mathcal{D}_c'|} \sum_{(\mathbf{X}_i, y_i) \in \mathcal{D}_c'} [\|f(\tau_m(\mathbf{X}_i; -\phi_m^c)) - f(\mathbf{X}_i)\|_1 + \|f(\tau_m(\mathbf{X}_i; \phi_m^c)) - f(\mathbf{X}_i)\|_1] \quad (2)$$

where $\tau_m(\mathbf{X}_i; -\phi_m^c)$ and $\tau_m(\mathbf{X}_i; \phi_m^c)$ are the transformed images of $\mathbf{X}_i$ by $-\phi_m^c$ and ϕ_m^c degrees, respectively, using the m-th transformation operation. These two are the maximum transformations that can be obtained by the m-th transformation operation given the hyperparameter ϕ_m^c. Because the objective function $\mathcal{J}_m$ has a complex form with respect to ϕ_m^c, we use Bayesian optimization to efficiently solve the optimization problem in Equation 1.

If $m \in \{8, 9, 10\}$, the search space for the hyperparameter ϕ_m^c is discrete and contains only two values: 0 and 0.5. Then, the function Δ_m evaluates the change in the softmax response when the transformation is applied, as follows:

$$\Delta_m(\phi_m^c; f, \mathcal{D}_c^{'}) = \frac{1}{|\mathcal{D}_c'|} \sum_{(\mathbf{X}_i, y_i) \in \mathcal{D}_c'} \|f(\tau_m(\mathbf{X}_i; 2 \cdot \phi_m^c)) - f(\mathbf{X}_i)\|_1 \quad (3)$$

where $\tau_m(\mathbf{X}_i; 2 \cdot \phi_m^c)$ indicates that the m-th transformation operation is applied to image $\mathbf{X}_i$ with a probability of $2 \cdot \phi_m^c$. Because the hyperparameter ϕ_m^c can have a value of either 0 or 0.5, we optimized it by directly comparing $\Delta_m(0; f, \mathcal{D}_c^{'})$ and $\Delta_m(0.5; f, \mathcal{D}_c^{'})$.

After solving the optimization problems for all pairs of transformation operations and classes, we obtain optimized hyperparameters of M transformation operations to specify class-specific augmentation policies $\Phi = \{\boldsymbol{\phi}^1, \ldots, \boldsymbol{\phi}^C\}$, where $\boldsymbol{\phi}^c = (\phi_1^c, \ldots, \phi_M^c)$. The c-th hyperparameter set $\boldsymbol{\phi}^c$ is used to specify the distributions of transformation operations for the class-specific augmentation policy $\mathcal{A}(\boldsymbol{\phi}^c)$. For class-adaptive data augmentation, an image $\mathbf{X}$ belonging to the c-th class is randomly transformed by applying a transformation τ sampled from $\mathcal{A}(\boldsymbol{\phi}^c)$ as $\tilde{\mathbf{X}} = \mathbb{E}_{\tau \sim \mathcal{A}(\boldsymbol{\phi}^c)}[\tau(\mathbf{X})]$.

An important consideration is the choice of the tolerance hyperparameter ϵ. If it is set to larger, the hyperparameters of transformation operations will become larger. Because the augmentation policy uses multiple transformation operations simultaneously, an image can be extremely transformed in a such way that its class label may not be preserved. However, setting ϵ to a smaller value results in smaller hyperparameters of transformation operations, and therefore, the effect of data augmentation will be reduced. To sidestep this difficulty, we set ϵ to be sufficiently large and introduce an augmentation decay strategy for using class-adaptive data augmentation in the next stage.

C. TRAINING WITH CLASS-ADAPTIVE DATA AUGMENTATION

In this stage, we re-train the CNN f with data augmentation using class-specific augmentation policies derived in the previous stage. At each training epoch, each training image $\mathbf{X}$ is transformed differently depending on its class label y using the class-specific augmentation policy $\mathcal{A}(\boldsymbol{\phi}^y)$. For example, an image $\mathbf{X}$ belonging to the c-th class is transformed using a transformation $\tau \sim \mathcal{A}(\boldsymbol{\phi}^c)$. The CNN uses a transformed image $\tilde{\mathbf{X}}$ to output its softmax response $f(\tilde{\mathbf{X}})$.

For class-adaptive data augmentation, we introduce an augmentation decay strategy that initially uses relatively large values for the hyperparameters $\Phi = \{\boldsymbol{\phi}^1, \ldots, \boldsymbol{\phi}^C\}$ and gradually decreases these values over time such that the gap between original and transformed images will be reduced as the training progresses [25]. To decrease the hyperparameter values, we decay the hyperparameters $\Phi = \{\boldsymbol{\phi}^1, \ldots, \boldsymbol{\phi}^C\}$ by a factor of $\gamma \in [0, 1)$ at every training epoch. With the augmentation decay, the CNN f is trained with intensively transformed images at early training epochs and is refined with fewer transformed images at later stages. This can contribute to better regularization of the CNN f [25].

IV. EXPERIMENTS

A. DATA DESCRIPTION

We validated the effectiveness of the proposed method using three benchmark image datasets with class-specific constraints of transformations: MNIST [26], SVHN [27], and WM-811K [28].

- **MNIST**: The dataset contains handwritten digit images, each belonging to one of the digits from 0 to 9. The original shape of each image is 28×28. In the experiments, we resized each image using bilinear interpolation and converted it into three channels, resulting in a shape of $32 \times 32 \times 3$. For data augmentation, each class has different restrictions on the rotation and flip operations owing to the characteristics of digits.
- **SVHN**: The dataset contains printed digit images cropped from images of house number plates. There are 10 digit classes from 0 to 9. Every image has a shape of $32 \times 32 \times 3$. For data augmentation, shear and color invert operations are known to be effective for all classes [10]. Each class has different restrictions on the rotation and flip operations, such as MNIST.
- **WM-811K**: The dataset contains semiconductor wafer bin maps with 9 defect pattern classes: *Center*, *Donut*, *Edge-Loc*, *Edge-Ring*, *Loc*, *Random*, *Scratch*, *Near-Full*, and *None*. In a wafer map, each element has a value of 1 if the corresponding die is defective and 0 otherwise. Among the total 811,457 wafer maps, 172,946 labeled wafer maps with a size greater than 100 were used in the experiment. Because the wafer maps were differently shaped, we resized them using bicubic interpolation and duplicated the channels to be 64×64. For data augmentation, the rotation and flip operations are label-preserving transformations for all

TABLE 2: Description of benchmark datasets

Dataset	Class Description	No. Training/Validation Images	No. Test Images	Original Image Shape	Resized Image Shape
MNIST	10 handwritten digits	60,000	10,000	(28, 28, 1)	(32, 32, 3)
SVHN	10 printed digits	73,257	26,032	(32, 32, 3)	(32, 32, 3)
WM-811K	9 wafer map defect patterns	54,354	118,592	(6, 21, 1) $\sim$ (300, 202, 1)	(64, 64, 3)

classes [5]. However, shift and zoom operations do not preserve labels for most classes.

Table 2 lists the dataset details. For each dataset, we used the default split of the training/validation and test sets in the original release. All pixel values were normalized using the mean and standard deviation of the training set.

B. EXPERIMENTAL SETTINGS

For the architecture of the CNN f, we used VGG19 [29] with batch normalization [30] and ResNet34 [31]. For each of them, we removed all fully-connected layers and added an output layer having as many nodes as the number of classes in the target image classification task. In the case of VGG19, we replaced the flatten operation with global average pooling.

In the first and third stages of the proposed method, we used the following configurations to train the CNN f. The ratio of training set $\mathcal{D}$ and validation set $\mathcal{D}^{'}$ was set to 8:2. Categorical cross-entropy was used as the loss function. We used the Adam optimizer for parameter updating, with a mini-batch size of 128. L2 regularization with a factor of 10^{-4} was applied to the parameters. The augmentation decay γ for the third stage was set to 0.98. The initial learning rate was set to 10^{-4}. The learning rate was reduced by a factor of 0.1 if the validation accuracy did not improve during 25 successive training epochs. After 100 training epochs, training was terminated if the validation accuracy did not improve during 50 successive epochs or the number of epochs reached 500.

In the second stage of the proposed method, we used the 10 transformation operations listed in Table 1 to form one class-specific augmentation policy. We set $\epsilon = 0.1$ and $\lambda = 0.01$ in the objective function $\mathcal{J}_m$. For Bayesian optimization, we used the Gaussian process with the Matern kernel as the surrogate model and expected improvement as the acquisition function. The numbers of initial random exploration steps and optimization steps were set to 5 and 30, respectively.

The CNN performance was evaluated in terms of the classification accuracy of the test set, which is a typically used performance measure in image classification tasks. All experiments were repeated 10 times independently with different random seeds, and the mean and standard deviation of each result over the 10 replications were calculated.

C. COMPARED METHODS

The proposed method, referred to as **ClassAdaptive-DA**, was compared with three baselines: CNN trained without data augmentation (**Without-DA**), CNN trained with data augmentation based on a known dataset-level augmentation policy in the literature (**Baseline-DA**), and an ablation of the proposed method in which the same augmentation policy applied to all classes instead of class-adaptive policies (**DatasetAdaptive-DA**).

For **Baseline-DA**, the augmentation policies for the MNIST, SVHN, and WM-811K datasets were obtained from previous studies [8], [10], and [5], respectively. The other training configurations were the same as those in the proposed method.

For **DatasetAdaptive-DA**, we altered the second stage of the proposed method by optimizing the dataset-level augmentation policy. This was to examine the effectiveness of class-adaptive augmentation policies in the proposed method while controlling other conditions.

D. RESULTS AND DISCUSSION

Table 3 presents a comparison of the classification accuracies of the baseline and proposed methods with two CNN architectures, VGG19 and ResNet34, on the benchmark datasets. In each row, the highest accuracy is shown in bold. The results show that **ClassAdaptive-DA** yielded a classification accuracy comparable to or superior to that of the baseline methods in most experimented cases. **Without-DA** always yielded the lowest classification accuracy. In the case of MNIST, wherein all methods yielded very high classification accuracy, **ClassAdaptive-DA** yielded the highest classification accuracy; however, there were no significant performance differences between **Baseline-DA**, **DatasetAdaptive-DA**, and **ClassAdaptive-DA**. In the case of SVHN, **DatasetAdaptive-DA** performed the best, followed by **ClassAdaptive-DA**. In the case of WM-811K, **ClassAdaptive-DA** outperformed all the baseline methods. Regarding the CNN architecture, VGG19 showed slightly better performance than ResNet34.

For **ClassAdaptive-DA**, we examined class-specific augmentation policies derived using VGG19 for each benchmark dataset. The optimized hyperparameters for MNIST, SVHN, and WM-811K are listed in Table 4, Table 5 and Table 6, respectively. The notable results in the tables are highlighted in bold. Figure 2 shows examples of class-adaptively transformed images at the 50-th training epoch of VGG19 for the benchmark datasets. We found the the class-specific augmentation policies derived using **ClassAdaptive-DA** were consistent with the known domain knowledge about each benchmark dataset. In addition, we found other class-specific

TABLE 3: Performance comparison on benchmark datasets in terms of classification accuracy (average±standard deviation)

Dataset	CNN Architecture	**Without-DA**	**Baseline-DA**	**DatasetAdaptive-DA** (ablation)	**ClassAdaptive-DA** (proposed)
MNIST	VGG19	99.519±0.035	99.623±0.048	99.582±0.030	**99.640±0.029**
	ResNet34	99.441±0.045	99.579±0.061	99.580±0.062	99.545±0.060
SVHN	VGG19	94.217±0.159	96.135±0.211	**96.982±0.147**	96.699±0.068
	ResNet34	91.911±0.280	94.276±0.176	95.243±0.136	94.675±0.169
WM-811K	VGG19	96.299±0.189	96.371±0.211	96.555±0.223	**96.727±0.212**
	ResNet34	96.082±0.147	96.149±0.208	96.495±0.275	96.514±0.181

label-preserving transformations that have not been discussed in the literature. Our findings for each dataset are described below.

- **MNIST**: The class-specific augmentation policies of MNIST are determined according to the characteristics of the digits. For rotation, a full range was used for class 0 only, whereas a smaller range was used for other classes. Wide shear ranges were used for all classes. Shear is a label-preserving transformation of handwritten digits as they are naturally sheared. For horizontal and vertical shifts, almost the full range was used for class 1 only. The probability of horizontal flip was always set to maximum for classes 0, 1, and 8, whereas horizontal flip was not used for classes ‘2, 3, 5, 6, and 9. The probability of vertical flip was maximum for classes 0, 1, 3, and 8, whereas vertical flip was not used for classes 2, 5, 6, 7, and 9. For class 4, horizontal and vertical flips had a non-zero probability. For certain handwritten four digits in the dataset, these flips appeared to preserve the labels.
- **SVHN**: The class-specific augmentation policies of SVHN were similar to those of MNIST. Additionally, the characteristics of the digits printed on the home number plates were observed. Compared with MNIST, the differences in the optimized hyperparameters between classes were relatively small, which may have degraded the effect of class-adaptive augmentation policies on classification accuracy. Some interesting characteristics different from those of MNIST are as follows: The ranges of the shear were set to be relatively narrow as the printed digits were generally less sheared than handwritten digits. Because the images in this dataset had various colors, color invert was label-preserving and was thus used with a wide range for all classes.
- **WM-811K**: The class-specific augmentation policies of WM-811K show the rotation-invariance properties of the wafer map defect patterns. A full range of rotation and maximum probability of horizontal/vertical flips were used for each class. Shear, shifts, and zoom were not label-preserving transformations for the wafer maps of specific classes. A small range of shear was used for classes “Edge-Loc” and “Edge-Ring” because this operation changed the edge location. Small ranges of horizontal/vertical shifts were used for classes “Center,” “Donut,” “Edge-Loc,” and “Edge-Ring,” because these operations changed the center and edge locations in a wafer map. A small range of zoom was used for classes “Edge-Loc,” “Edge-Ring,” and “Near-Full” because this operation changed the edge location. Color invert was only applied to class Random.

V. CONCLUSION

We presented a class-adaptive data augmentation method for effectively using data augmentation in image classification tasks in the absence of domain knowledge of a given dataset. Using a CNN trained without data augmentation, class-specific augmentation policies for individual classes were derived by performing an optimization procedure. The CNN was re-trained using data augmentation with class-specific augmentation polices. Experimental results using benchmark datasets with class-specific characteristics showed that the proposed method yielded comparable or superior performance to the baseline methods. Additionally, it was shown that the class-specific augmentation policies derived using the proposed method for each dataset were consistent with the domain knowledge.

We expect that the proposed method will be useful in further improving the classification accuracy of a CNN when class-specific constraints of transformations are imposed on the dataset. The proposed method requires maintaining only one CNN, which is advantageous in terms of the memory usage. Additionally, the analysis of class-specific augmentation policies can aid in understanding the class-specific characteristics of a dataset without domain knowledge.

In future work, we will investigate methods to further improve the efficiency and effectiveness of the proposed method. The efficiency can be improved by training the CNN and optimizing class-specific augmentation policies simultaneously. The effectiveness can be improved by introducing more randomness into augmentation policies, such as randomly permuting the order between transformation operations and randomly sampling transformation operations to apply, to generate more diversified variations of images.

REFERENCES

[1] H. Inoue, “Data augmentation by pairing samples for images classification,” arXiv preprint, vol. arXiv:1801.02929, 2018.
[2] A. Mikołajczyk and M. Grochowski, “Data augmentation for improving deep learning in image classification problem,” in Proceedings of International Interdisciplinary PhD Workshop, 2018, pp. 117–122.
[3] C. N. Vasconcelos and B. N. Vasconcelos, “Convolutional neural network committees for melanoma classification with

TABLE 4: Optimized hyperparameters of class-specific augmentation policies for VGG19 on MNIST dataset

Class	Rotation (ϕ_1)	Shear (ϕ_2)	Horizontal Shift (ϕ_3)	Vertical Shift (ϕ_4)	Zoom (ϕ_5)	Brightness (ϕ_6)	Contrast (ϕ_7)	Color Invert (ϕ_8)	Horizontal Flip (ϕ_9)	Vertical Flip (ϕ_{10})
0	**90.00±0.00**	**45.00±0.00**	15.08±1.39	15.84±1.92	49.39±1.84	50.00±0.00	50.00±0.00	0.00±0.00	**0.50±0.00**	**0.50±0.00**
1	53.46±4.63	**45.00±0.00**	**50.00±0.00**	**48.02±5.95**	50.00±0.00	50.00±0.00	50.00±0.00	0.25±0.25	**0.50±0.00**	**0.50±0.00**
2	44.80±1.60	**37.87±1.34**	30.52±3.03	18.20±1.70	50.00±0.00	50.00±0.00	50.00±0.00	0.00±0.00	0.00±0.00	0.00±0.00
3	49.93±2.22	**43.01±1.81**	27.50±3.82	23.35±3.11	49.17±1.58	50.00±0.00	50.00±0.00	0.05±0.15	0.00±0.00	**0.50±0.00**
4	39.18±1.87	**37.75±2.06**	24.76±3.49	21.41±3.57	50.00±0.00	50.00±0.00	50.00±0.00	0.05±0.15	**0.40±0.20**	0.20±0.24
5	55.41±2.94	**44.13±1.43**	28.94±3.06	18.34±2.22	50.00±0.00	50.00±0.00	50.00±0.00	0.30±0.24	0.00±0.00	0.00±0.00
6	50.03±2.16	**43.27±2.02**	16.63±3.50	16.93±3.03	49.60±1.21	50.00±0.00	50.00±0.00	0.00±0.00	0.00±0.00	0.00±0.00
7	38.36±2.11	**37.03±1.90**	27.17±3.62	15.20±0.85	50.00±0.00	50.00±0.00	50.00±0.00	0.00±0.00	0.05±0.15	0.00±0.00
8	52.48±2.44	**45.00±0.00**	24.07±3.35	17.05±1.22	50.00±0.00	50.00±0.00	50.00±0.00	0.40±0.20	**0.50±0.00**	**0.50±0.00**
9	55.56±3.43	**43.87±1.58**	21.84±3.44	16.75±2.14	50.00±0.00	50.00±0.00	50.00±0.00	0.30±0.24	0.00±0.00	0.00±0.00

TABLE 5: Optimized hyperparameters of class-specific augmentation policies for VGG19 on SVHN dataset

Class	Rotation (ϕ_1)	Shear (ϕ_2)	Horizontal Shift (ϕ_3)	Vertical Shift (ϕ_4)	Zoom (ϕ_5)	Brightness (ϕ_6)	Contrast (ϕ_7)	Color Invert (ϕ_8)	Horizontal Flip (ϕ_9)	Vertical Flip (ϕ_{10})
0	37.07±2.76	**34.16±3.46**	19.01±1.12	28.53±1.60	49.13±2.00	50.00±0.00	50.00±0.00	**0.50±0.00**	**0.50±0.00**	**0.50±0.00**
1	35.27±2.26	**29.20±2.51**	**50.00±0.00**	**37.47±4.42**	50.00±0.00	50.00±0.00	50.00±0.00	**0.50±0.00**	**0.50±0.00**	**0.50±0.00**
2	31.38±2.25	**25.14±1.34**	21.56±1.08	24.71±1.08	48.76±1.79	50.00±0.00	50.00±0.00	**0.50±0.00**	0.00±0.00	0.00±0.00
3	28.73±2.50	**25.15±1.26**	21.32±1.16	26.33±1.73	49.40±1.20	50.00±0.00	50.00±0.00	**0.50±0.00**	0.00±0.00	**0.50±0.00**
4	26.68±1.45	**25.76±6.56**	20.43±0.75	26.28±0.50	50.00±0.00	50.00±0.00	50.00±0.00	**0.50±0.00**	**0.50±0.00**	0.00±0.00
5	30.93±3.35	**24.66±1.32**	20.50±0.94	26.84±1.01	49.84±0.41	50.00±0.00	50.00±0.00	**0.50±0.00**	0.00±0.00	0.00±0.00
6	32.15±2.55	**26.26±1.67**	19.95±0.64	23.92±1.11	44.88±2.41	50.00±0.00	50.00±0.00	**0.50±0.00**	0.00±0.00	0.00±0.00
7	24.35±1.97	**23.46±1.46**	22.44±1.19	24.77±1.42	50.00±0.00	50.00±0.00	50.00±0.00	**0.50±0.00**	0.00±0.00	0.00±0.00
8	31.62±2.36	**27.04±1.77**	18.99±1.03	23.40±0.98	44.80±2.58	50.00±0.00	50.00±0.00	**0.50±0.00**	**0.50±0.00**	**0.50±0.00**
9	33.19±3.30	**26.39±1.68**	19.70±0.55	23.02±1.13	44.53±3.11	50.00±0.00	50.00±0.00	**0.50±0.00**	0.00±0.00	0.00±0.00

TABLE 6: Optimized hyperparameters of class-specific augmentation policies for VGG19 on WM-811K dataset

Class	Rotation (ϕ_1)	Shear (ϕ_2)	Horizontal Shift (ϕ_3)	Vertical Shift (ϕ_4)	Zoom (ϕ_5)	Brightness (ϕ_6)	Contrast (ϕ_7)	Color Invert (ϕ_8)	Horizontal Flip (ϕ_9)	Vertical Flip (ϕ_{10})
Center	**90.00±0.00**	45.00±0.00	**7.46±0.74**	**7.97±0.86**	50.00±0.00	49.77±0.68	49.81±0.58	0.00±0.00	**0.50±0.00**	**0.50±0.00**
Donut	**90.00±0.00**	35.78±1.67	**10.29±6.64**	**13.77±6.03**	29.49±3.05	49.76±0.71	49.68±0.67	0.00±0.00	**0.50±0.00**	**0.50±0.00**
Edge-Loc	**90.00±0.00**	**20.23±1.38**	**9.41±0.99**	**10.50±1.19**	**8.28±0.82**	49.41±1.01	50.00±0.00	0.00±0.00	**0.50±0.00**	**0.50±0.00**
Edge-Ring	**90.00±0.00**	**16.89±1.87**	**5.89±1.22**	**7.07±0.65**	**7.80±0.63**	50.00±0.00	50.00±0.00	0.00±0.00	**0.50±0.00**	**0.50±0.00**
Loc	**90.00±0.00**	45.00±0.00	26.47±1.88	28.22±1.77	43.34±2.40	49.16±1.38	49.72±0.60	0.00±0.00	**0.50±0.00**	**0.50±0.00**
Random	**90.00±0.00**	39.56±2.75	36.04±3.43	35.99±3.47	25.83±2.23	41.79±4.73	49.79±0.63	**0.50±0.00**	**0.50±0.00**	**0.50±0.00**
Scratch	**90.00±0.00**	45.00±0.00	43.21±5.57	44.85±3.97	44.59±4.63	50.00±0.00	50.00±0.00	0.00±0.00	**0.50±0.00**	**0.50±0.00**
Near-Full	**90.00±0.00**	30.38±2.39	23.25±4.40	21.94±3.28	**14.61±2.09**	49.38±0.98	50.00±0.00	0.00±0.00	**0.50±0.00**	**0.50±0.00**
None	**90.00±0.00**	45.00±0.00	50.00±0.00	50.00±0.00	50.00±0.00	50.00±0.00	50.00±0.00	0.00±0.00	**0.50±0.00**	**0.50±0.00**

classical and expert knowledge based image transforms data augmentation," arXiv preprint, vol. arXiv:1702.07025, 2017.

[4] G. Wimmer, A. Uhl, and A. Vecsei, "Evaluation of domain specific data augmentation techniques for the classification of celiac disease using endoscopic imagery," in Proceedings of IEEE International Workshop on Multimedia Signal Processing, 2017.

[5] S. Kang, "Rotation-invariant wafer map pattern classification with convolutional neural networks," IEEE Access, vol. 8, pp. 170 650–170 658, 2020.

[6] H. Hu, C. He, and P. Li, "Semi-supervised wafer map pattern recognition using domain-specific data augmentation and contrastive learning," in Proceedings of IEEE International Test Conference, 2021, pp. 113–122.

[7] H. Lee and J. Lee, "Neural network prediction of sound quality via domain knowledge-based data augmentation and bayesian approach with small data sets," Mechanical Systems and Signal Processing, vol. 157, p. 107713, 2021.

[8] P. Y. Simard, D. Steinkraus, and J. C. Platt, "Best practices for convolutional neural networks applied to visual document analysis," in Proceedings of International Conference on Document Analysis and Recognition, 2003, pp. 958–963.

[9] C. Lei, B. Hu, D. Wang, S. Zhang, and Z. Chen, "A preliminary study on data augmentation of deep learning for image classification," in Proceedings of Asia-Pacific Symposium on Internetware, 2019.

[10] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le, "AutoAugment: Learning augmentation strategies from data," in Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 113–123.

[11] D. Ho, E. Liang, X. Chen, I. Stoica, and P. Abbeel, "Population based augmentation: Efficient learning of augmentation policy schedules," in Proceedings of International Conference on Machine Learning, 2019, pp. 2731–2741.

[12] S. Lim, I. Kim, T. Kim, C. Kim, and S. Kim, "Fast AutoAugment," in Advances in Neural Information Processing Systems, 2019, pp. 6665—-6675.

[13] X. Zhang, Q. Wang, J. Zhang, and Z. Zhong, "Adversarial AutoAugment," in Proceedings of International Conference on Learning Representations, 2020.

[14] R. Hataya, J. Zdenek, K. Yoshizoe, and H. Nakayama, "Faster AutoAugment: Learning augmentation strategies using backpropagation," in Proceedings of European Conference on Computer Vision, 2020.

[15] Y. Li, G. Hu, Y. Wang, T. Hospedales, N. Robertson, and Y. Yang, "Differentiable automatic data augmentation," in Proceedings of European Conference on Computer Vision, 2020, pp. 580–595.

[16] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," Journal of Big Data, vol. 6, no. 1, pp. 1–48, 2019.

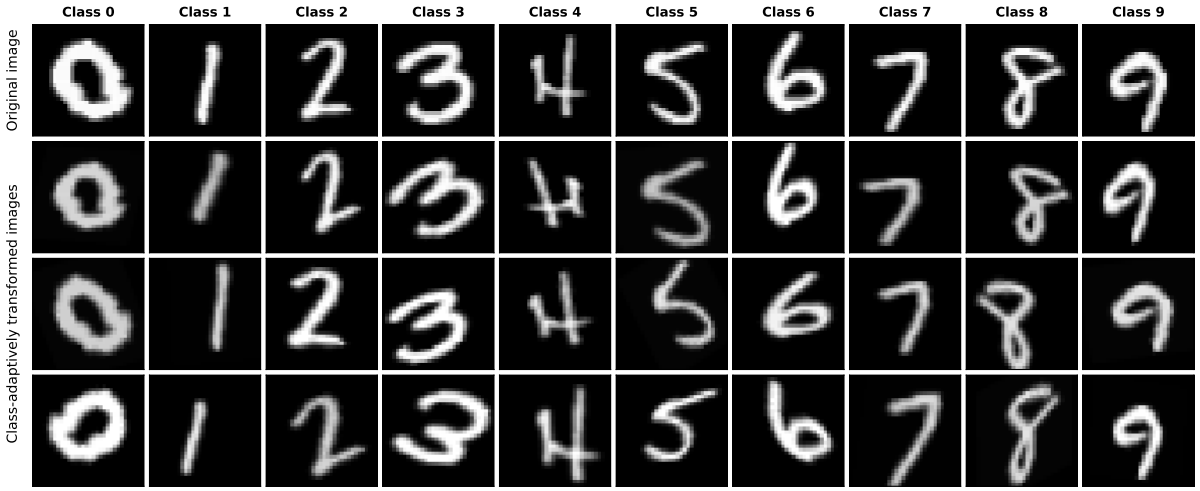
[17] L. Perez and J. Wang, "The effectiveness of data augmentation in image classification using deep learning," arXiv preprint, vol. arXiv:1712.04621, 2017.

[18] F. López de la Rosa, J. L. Gómez-Sirvent, R. Sánchez-Reolid, R. Morales, and A. Fernández-Caballero, "Geometric transformation-based data augmentation on defect classification of segmented images of semiconductor materials using a ResNet50 convolutional neural network," Expert Systems with Applications, vol. 206, p. 117731, 2022.

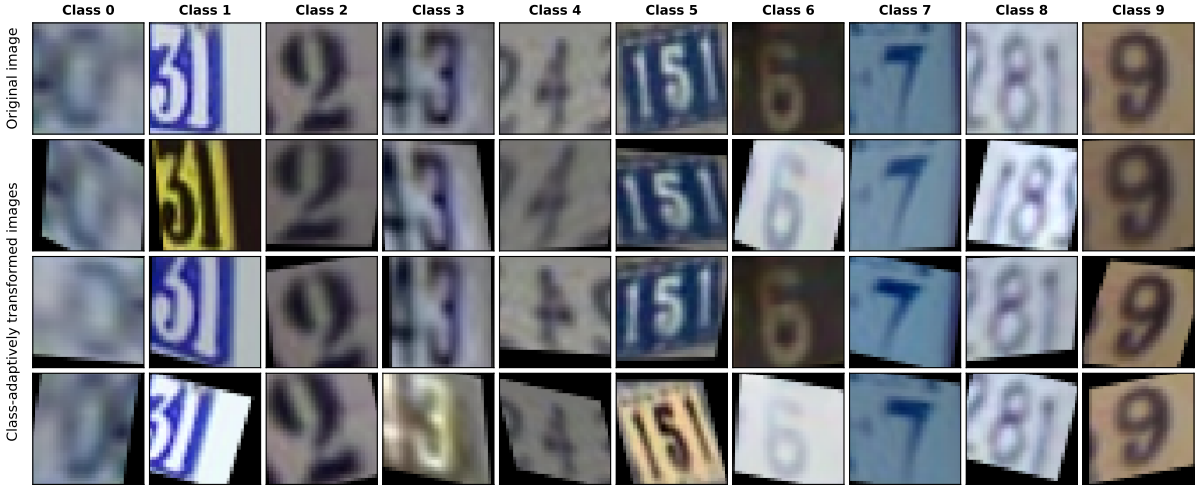
[19] L. Taylor and G. Nitschke, "Improving deep learning with generic data augmentation," in Proceedings of IEEE Symposium Series on Computational Intelligence, 2018, pp. 1542–1547.

[20] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "MixUp: Beyond empirical risk minimization," in Proceedings of International Conference on Learning Representations, 2018.

(a) MNIST dataset



(b) SVHN dataset



(c) WM-811K dataset

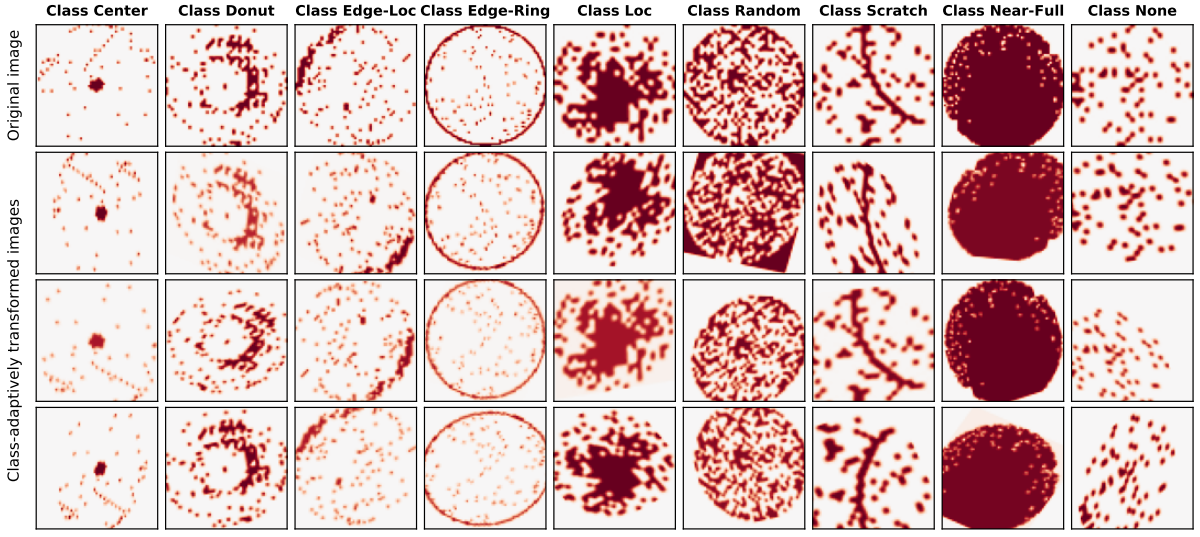


FIGURE 2: Examples of class-adaptively transformed images at 50-th training epoch of VGG19

[21] S. Yun, D. Han, S. J. Oh, S. Chun, J. Choe, and Y. Yoo, "CutMix: Regularization strategy to train strong classifiers with localizable features," in Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 6023–6032.
[22] C.-L. Li, K. Sohn, J. Yoon, and T. Pfister, "CutPaste: Self-supervised learning for anomaly detection and localization," in Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 9664–9674.
[23] F. Zhou, J. Li, C. Xie, F. Chen, L. Hong, R. Sun, and Z. Li, "MetaAugment: Sample-aware data augmentation policy learning," in Proceedings of AAAI Conference on Artificial Intelligence, 2021, pp. 11 097–11 105.
[24] T.-H. Cheung and D.-Y. Yeung, "AdaAug: Learning class- and instance-adaptive data augmentation policies," in Proceedings of International Conference on Learning Representations, 2022.
[25] Z. He, L. Xie, X. Chen, Y. Zhang, Y. Wang, and Q. Tian, "Data augmentation revisited: Rethinking the distribution gap between clean and augmented data," arXiv preprint, vol. arXiv:1909.09148, 2019.
[26] L. Deng, "The MNIST database of handwritten digit images for machine learning research," IEEE Signal Processing Magazine, vol. 29, no. 6, pp. 141–142, 2012.
[27] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in Advances in Neural Information Processing Systems, 2011.
[28] M. J. Wu, J. S. R. Jang, and J. L. Chen, "Wafer map failure pattern recognition and similarity ranking for large-scale data sets," IEEE Transactions on Semiconductor Manufacturing, vol. 28, no. 1, pp. 1–12, 2014.
[29] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in Proceedings of International Conference on Learning Representations, 2015.
[30] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating deep network training by reducing internal covariate shift," in Proceedings of International Conference on Machine Learning, 2015, pp. 448–456.
[31] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.

PLACE PHOTO HERE

JISU YOO received the B.S. degrees in industrial and systems engineering and in software convergence from Dongguk University in 2021. He is currently a M.S. student at the Department of Industrial Engineering, Sungkyunkwan University. His research interests include data augmentation, hyperparameter optimization, and industrial arificial intelligence.

SEOKHO KANG is an assistant professor of industrial engineering at Sungkyunkwan University. He received the B.S. and Ph.D. degrees in industrial engineering from Seoul National University in 2011 and 2015, respectively. He was a research staff member with Samsung Advanced Institute of Technology. His research focuses on bridging the gap between theory and practice in machine learning through practical methodologies. He is also interested in data mining applications in various industrial fields, incldling manufacturing, materials, and healthcare.