Pattern Recognition 127 (2022) 108637

Contents lists available at ScienceDirect

Pattern Recognition

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

GA3N: Generative adversarial AutoAugment network

Vanchinbal Chinbat[a,1], Seung-Hwan Bae[b,*]

[a] Incheon National University, South Korea
[b] Inha University, South Korea

ARTICLE INFO

Article history:
Received 6 August 2021
Revised 3 January 2022
Accepted 7 March 2022
Available online 9 March 2022

Keywords:
Data augmentation
AutoAugment
Generative adversarial network
Classification
Deep learning
Adversarial learning

ABSTRACT

Data augmentation is beneficial for improving robustness of deep meta-learning. However, data augmentation methods for the recent deep meta-learning are still based on photometric or geometric manipulations or combinations of images. This paper proposes a generative adversarial autoaugment network (GA3N) for enlarging the augmentation search space and improving classification accuracy. To achieve, we first extend the search space of image augmentation by using GANs. However, the main challenge is to generate images suitable for the task. For solution, we find the best policy by optimizing a target and GAN losses alternatively. We then use the manipulated and generated samples determined by the policy network as augmented samples for improving the target tasks. To show the effects of our method, we implement classification networks by combining our GA3N and evaluate them on CIFAR-100 and Tiny-ImageNet datasets. As a result, we achieve better accuracy than the recent AutoAugment methods on each dataset.

1. Introduction

Image manipulation is beneficial for increasing the size of samples and features. Many works [1–4] for image classification have shown that exploiting this method improves the generalization ability of the trained model. However, the fixed augmentation policies for dealing with the specific target tasks [5,6] do not transfer well to other tasks in general. Therefore, selecting suitable operations automatically according to a target task is indeed important to transfer a model to another domain.

As one of the pioneer works, AutoAugment [5] searches and selects augmentation operations automatically for a proxy task (i.e., smaller models and reduced dataset), and then applies the selected operations for the classification data set with much more samples.

However, the computational cost of training and evaluating thousands of sampled policies is a burden when applying this method for large-scale tasks. In addition, the operations optimized on the proxy task are not likely to be optimal operations on the target task. To resolve the drawbacks of AutoAugement, the adversarial AutoAugment method [7] has been presented. Rather than finding appropriate policies by investigating huge operations on proxy tasks beforehand, it tries to find the optimal operations by using the adversarial loss. As a result, it can train a target model on full datasets because the overfitting problem from the redundant samples is alleviated by generating hard samples.

On the other hand, the adversarial AutoAugment [7] maintains the search space of operations used in AutoAugment [5]. The search space contains 16 geometric and photometric operations. This means that the scalability and generalization of the AutoAugment network can be limited due to the fixed search space. One of the remedies is to leverage synthetic samples generated by GANs. In [8,9], GANs can be used for data augmentation and is helpful to resolve the target tasks. However, a simple approach to generate a large number of synthetic samples, and use these as training samples is not effective. Moreover, joint training between GANs and target networks is needed to generate meaningful and difficult samples for the target network. As a result, a unified framework eligible for joint training among a target, policy, and generation networks is required to improve the scalability and generalization ability of the AutoAugment method further.

To achieve this, we propose a novel generative adversarial AutoAugment method. The main idea is to enlarge policy search space by adding new GAN operations with conditional GAN (CGAN) [10] and data augmentation GAN (DAGAN) [6]. Therefore, we can enlarge the policy space by using 16 image and GAN operations. We then find optimal policies by adversarial training between a target network and a policy controller network. Subsequently, we generate various samples according to the selected operations and use them for training a target network. Based

* Corresponding author.
E-mail addresses: vansanety@gmail.com (V. Chinbat), shbae@inha.ac.kr (S.-H. Bae).

[1] This work was done while V. Chinbat was studying at vision and learning Lab.@Inha University.

https://doi.org/10.1016/j.patcog.2022.108637
0031-3203/© 2022 Elsevier Ltd. All rights reserved.

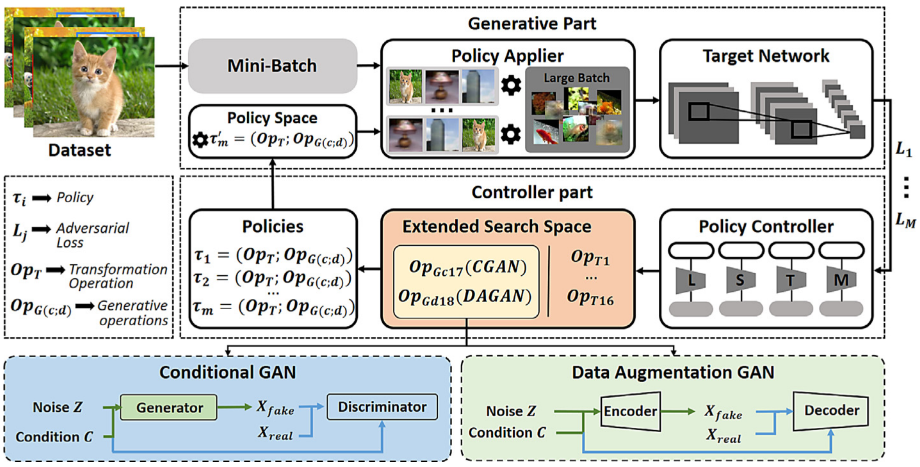

Fig. 1. The proposed framework of GA3N. In the generative part, the policy applier generates a training batch with augmented samples by applying update sample policies to image samples, and a target network is trained with the batch by minimizing a target loss. On the other hand, the policy controller predicts new augmentation policies by maximizing the target loss at the next iteration in the controller part. Here, CGAN and DAGAN are added in the policy search space for generating new synthetic images.

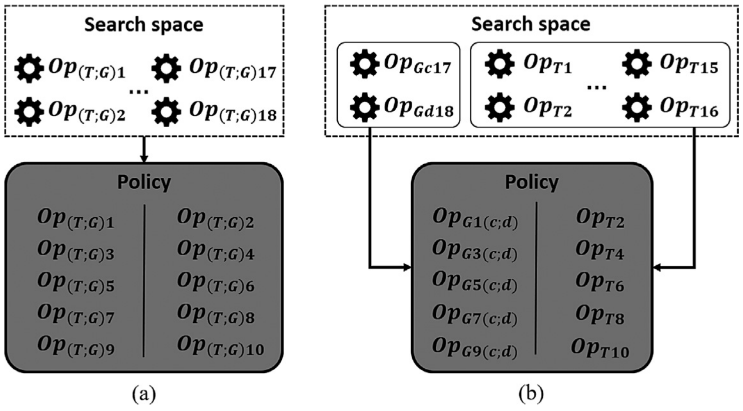

Fig. 2. Two augmentation policy search spaces by using synthetic image generators as augmentation operators. Scenario (a) represents the controller-dependent search for 18 operations including CGAN Op_{Gc17} and DAGAN Op_{Gd18}. On the other side, scenario (b) illustrates the GAN-dependent search that forcing one of the operations of sub-policy to be one of the generative models $Op_{G(c;d)}$.

on this idea, we efficiently combine the adversarial AutoAugment with the recent GANs and present a GA3N framework shown in Fig. 1.

For training GA3N, we pre-train CGAN [11] and DAGAN [6] on the given datasets to learn the data distribution of the given images. Then, we construct a policy search space with 16 image manipulations and these two synthetic image generation operations of GANs (in total 18 operations). For the extended policy space, we find optimal operations via the adversarial training between the policy controller and target networks. During this adversarial training, the target network can be trained by minimizing the task loss of the target task which is evaluated with the real and augmented samples together. On the other hand, the policy network can be trained by maximizing the loss in order to generate harder augmented samples at the next iteration. By feeding various samples to the target network, we can improve the robustness of this network. As a result, this adversarial training is beneficial for improving the generalization and discrimination abilities of both policy and target networks, respectively.

For the construction of a policy search space shown in Fig. 2, we present two kinds of operation selection methods: a controller-dependent selection and a GAN-dependent selection. In the former, each generative network is considered as one operation and contained into a unified policy search space with other operations. Each sub-policy consists of two operations selected by the predicted actions of one policy controller. Therefore, any GAN operation could not be contained in the augmentation policies. On the other hand, in the latter one, we force each sub-policy to contain at least one of the GAN operations. This can be achieved by dividing a policy space into two subspaces. Here, one sub-space contains GAN operations only, but the other contains conventional image manipulation operations. In addition, two individual controllers are trained to provide the proper actions for each subspace. According to the actions of both controllers, one operation can be selected independently in each subspace, and the selected two operations can compose a policy subset. We implement these controllers based on single-layer LSTM with a different numbers of LSTM units as shown in Fig. 5. In our evaluation, we compare these policy search methods on several tasks as shown in Section 5.4. For the evaluation on CIFAR-100 and Tiny-ImageNet, we improved the classification accuracy by 0.3% and 1.1% compared to the [7] and [12], respectively.

The contributions of this paper can be summarized as follows:

- Proposing a new adversarial AutoAugment framework with several GANs for automated data augmentation depending on a target task;
- Integrating two adversarial learning approaches into the unified framework to achieve the generation of both hard and synthetic samples;
- Presenting two different methods for constructing policy search space and their verification with the extensive ablation study;

- Presenting distinct improvement compared with recent AutoAugment methods [5,7] on CIFAR-100 and Tiny-ImageNet datasets.

2. Related work

Data augmentation: Since the quantity of data usually affects the performance of a target model, data augmentation is regarded as crucial in many research areas. As a simple augmentation method, [13,14] change the photometric and geometric properties of an image by applying several image manipulation techniques. In particular, they exploit kernel filters, geometric transformations or image mixing. Although each image manipulation method has the low computational complexity itself in general, the overall computational complexity of training a target model is not less because a bunch of augmented images should be generated and used for training a generalized network [13–15]. This problem is occurred due to the similar statistics of augmented images.

For more effective augmentation, some deep learning methods for data augmentation [16–19] have been developed. In particular, adversarial training between networks is key in these methods. There are two approaches using this adversarial training for data augmentation: (1) Synthetic image generation-based and (2) hard example mining-based approaches. The first one is to produce synthetic images using GANs [6,8,10,20–22]. By the adversarial training between a generator and a discriminator, they can produce large synthetic images and uses these ones as augmented images. The other approach is to determine the best set of combinations among many augmentation operations to reduce a loss of a target model. Therefore, the latter one is more focused on generating hard examples which increase the loss of a target model instead of generating synthetic images.

Synthetic image generation-based methods: In GANs, a generator struggles to fool a discriminator by generating synthetic images similar to real ones. Many works [8,10,11,23] proved that training a target network with extra synthetic images is effective to increase the accuracy in the many fields of computer vision [24]. The most representatives are object classification [5], detection [25], and segmentation [24]. Among these, the most studied task is object classification. Here, the variability of the training samples is the most important. From this perspective, the synthetic samples of GANs are useful to achieve that. One of the most successful examples is CycleGAN [8]. It reduces the discrepancy when translating an image from a source domain to a target domain with an adversarial loss. Furthermore, the Balancing GAN [21] uses GAN for data augmentation on imbalanced datasets. Although our GA3N and BAGAN harness GANs for improving target network accuracy, both methods are clearly different in the technical aspect: BAGAN focuses on generating minority-class images to make the balanced dataset, whereas our method generates the hard samples to deceive a target network.

In order to satisfy the generalization and robustness of a neural network, the quality and hardness of a training dataset are important. To this end, the methods for making the train samples harder than the original data is suggested for data augmentation [26], which is called the hard example-mining based method. In brief, this method generates more difficult samples that a target network handles by mining and combining existing samples. For this goal, some online-hard example mining methods [27] choose the hardest sample pairs in a mini-batch using a metric distance. In addition, semi-online mining methods [28–30] select sample pairs randomly from the hard enough sample pairs in a mini-batch.

In recent years, meta-learning [31], which uses the learned information from a task to improve a target task, has been introduced for hard example mining methods [5,7,32,33] called AutoAugment. The main idea of these methods is to optimize a target network with the feedback (i.e., augmentation policy) of the controller network. During the training iterations, the controller network produces harder augmentation policies to maximize the loss (*or* reward) of the target network. As one of pioneer works, smart augmentation [32] trains a target network by merging multiple images while the policy network tries to find the best combination of images with the loss of the target network. AutoAugment [5] uses a reinforcement algorithm to find an optimal augmentation policy among 16 augmentation operations. They define a policy by combining several operations and each policy is determined by an RNN-based controller network. Although the experimental results show a great improvement for the classification accuracy on several datasets, the training process is rather costly and complex due to the pre-training on proxy tasks.

To overcome this limitation, population-based augmentation (PBA) [33] present the dynamic scheduler network to learn the augmentation schedule rather than using the fixed augmentation process. However, this method still needs to pre-train the scheduler on the proxy dataset. For learning the controller network without the pre-training, the adversarial AutoAugment [7] presents the adversarial learning based on the Min-Max game of GANs. However, they use adversarial learning to determine the augmentation policy for training a target network. Therefore, the augmentation policy is still limited to the transformation of the existing training images.

In order to maximize the effects of the data augmentation, we design our GA3N based on two adversarial learning approaches. By incorporating CGAN [10] and DAGAN [6] into our framework as shown in Fig. 1, our GA3N can generate synthetic images. As an effect, the operation search space can be extended with added GAN operations. From the adversarial joint training between a target network and a policy controller, the policy network seeks to find augmentation operations which minimize the target loss. Therefore, a target network in our GA3N can be trained with synthetic and hard samples generated from these adversarial learning approaches. These contribute to improving the robustness and generalization of a target model.

3. Methodology

In this section, we present the methodology for our Generative Adversarial AutoAugment network. We first explain an overall framework of GA3N in Section 3.1. The motivation of the generative adversarial training between policy and target networks will be discussed in Section 3.2. Then, we present a method to extend the operation search space by adding the generative models in Section 3.3. Finally, the joint learning framework for target network training and augmentation policy search will be presented in Section 3.4.

3.1. Overall framework of GA3N

The proposed overall framework is shown in Fig. 1. The framework mainly consists of the generative and controller parts. We explain the components of each part as follows:

Generative part: The synthetic and the augmented hard samples are generated and provided to the target network. In the policy space, the sample policy is updated from the outputs of the policy controller and is fed to the policy applier one-by-one. The policy applier augments training samples by applying the updated policy for a given input sample. Then, a target network is trained with the larger batch containing real and augmented samples by minimizing its loss, and the loss of this network is fed to the policy controller.

Controller part: The policies within the policy search space are selected from the output of the policy controller. Here, the policy

Table 1
The list of the augmentation operations used for the extended search space. The top 16 image manipulation operations use the magnitude within the given range, but the GAN operations do not use the magnitude.

Operation Type	Operation Description
ShearX (Y)	Shearing the image horizontally (vertically) with the magnitude range of [-0.3, 0.3]
TranslateX (Y)	Translating the image horizontally (vertically) by the pixel magnitude range of [-150, 150]
Rotate	Rotating the image with the degree magnitude range of [-30, 30]
AutoContrast	Modifying the image contrast by maximizing the black and white pixels
Invert	Inverting the pixels of the image
Equalize	Equalizing the image's histogram
Solarize	Inverting the pixels above the threshold magnitude range of [0, 256]
Posterize	Reducing the each pixel's bits in the range of [4, 8]
Contrast	Modifying the contrast of the image in the range of [0.1, 1.9]
Color	Adjusting the color balance of the image in the magnitude range of [0.1, 1.9]
Brightness	Adjusting the brightness of the image in the magnitude range of [0.1, 1.9]
Sharpness	Adjusting the sharpness of the image in the magnitude range of [0.1, 1.9]
Cutout	Setting the square sized pixels into gray in the magnitude range of [0, 60]
Sample Pairing	Add a random image to another image from the same mini-batch in the magnitude range of [0, 0.4]
CGAN	Generative pretrained model for Conditional GAN
DAGAN	Generative pretrained model for Data Augmentation GAN

controller based on a single-layer LSTM can predict the operation type and magnitude in a sequential manner. The training of this controller network is achieved by maximizing the propagated loss from a target network. The synthetic generators of CGAN and DAGAN are added in the augmentation operation space. Here, each generator is also trained with the corresponding discriminator in an adversarial manner.

3.2. Motivation of two adversarial learning in GA3N

Adversarial learning [34] is one of the most powerful methods to make the target network optimized. In some sense, this is analogous to reinforcement learning since they adjust rewards at current states. As similar to GAN learning, which encourages a generator to produce realistic samples to deceive a discriminator, adversarial learning can be used to find optimal policies so that maximize the reward signal to a given dataset. This adversarial learning is named adversarial AutoAugment [7]. In general, the determined policies include augmentation operations to generate hard examples which is difficult to distinguish from a target network. For image classification, this training with hard samples is helpful to construct the more accurate decision regions for the training set.

On the other hand, it is also crucial to improve the generalization of a classification network. Although different combinations of manipulation operations determined by the adversarial AutoAugment increases the diversity of images for photometric and geometric variations, it cannot generate new images (but with similar distribution to training data). To achieve this, we use additional image generation operations of CGAN and DAGAN which is one of the main contributions of our paper. By our effective joint training scheme of these two adversarial learning methods as shown in Fig. 1 (i.e. adversarial learning between target and policy networks and generative adversarial networks), our GA3N can improve the generalization and discrimination abilities of a target network.

3.3. Extended search space - GAN integration

In AutoAugement, the search space is presumed to contain image operations to be applied for data augmentation during a target network training. A policy is constructed with the combinations of the operations. In this paper, we adopt the same search space with [7] for the traditional operations. We also omit a probability to select each operation (c.f., AutoAugment [5]) A policy contains 5 sub-policies, and each sub-policy contains two operations and the operation magnitudes. At initial training 5 sub-policies are chosen randomly within a search space, but determined by the outputs of the controller network at the next iteration.

In the implementation of [5] and [7], the search space includes 16 geometric and photometric image augmentation operations as shown in Table 1. In this work, we add two synthetic image generation operations using CGAN and DAGAN. CGAN can generate synthetic images depended on a given class label. By feeding the auxiliary information to both the generator and discriminator, CGAN can generate more class-specific images. Here, the auxiliary information can be any information (e.g. class label or data from other modalities). On the other hand, DAGAN can learn a model with larger invariance in different source domains. It first learns an embedding feature of input using a generator, and then learn a transformation function which can map from the low-dimensional feature to other within-class images. Therefore, we use CGAN and DAGAN as synthetic image generation operators because they are complementary to each other. Table 7 and 8 illustrate the impact of search space extension with each GAN.

By using synthetic images for the training of the target network, a training set gets diverse more and it improves the generalization ability of the network. Here, we present two policy search methods as shown in Fig. 2: Controller-dependent search and GAN-dependent search. In both searches, the search space is expanded for the policy possibilities from $|S| = (16 \times 10)^{10}$ [7] to $|S| = (16 \times 10 + 2)^{10}$ for the Controller-dependent search and $|S| = (16 \times 10)^5 + 2^5$ for the GAN-dependent searches. Here, $|S|$ is the search space scale of possible policies. Each of the 16 image manipulation operations has magnitudes discretized by 10 levels, but two CGAN and DAGAN operations do not use magnitude for image generation.

3.3.1. Controller-dependent search

For 18 operations in Table 1, the types and magnitudes of operations are determined with the predictions of the controller network. From the predictions, a single policy is constructed by concatenating 5 different sub-policies. Here, the sub-policy consisting of 2 operations are determined by the predicted results of the controller. In Fig. 3 and 4, augmented images from this controller-dependent search are shown.

3.3.2. GAN-dependent search

For using synthetic image generation operators more frequently, we make a sub-policy contain one image generation operation from either CGAN or DAGAN. To achieve this, we devise common and individual controllers for selecting GAN operations in different ways. We use a single LSTM network as the common controller. We select one of the non-GAN operations according to the predictions of the LSTM but select one of two GAN operations ran-

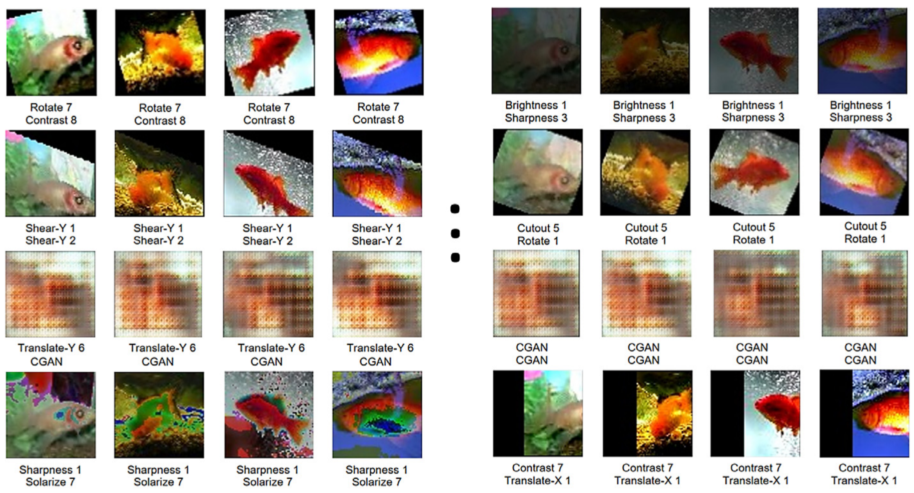

Fig. 3. Augmented images by applying each policy for a training image with a controller-dependent search on the Tiny-ImageNet dataset.

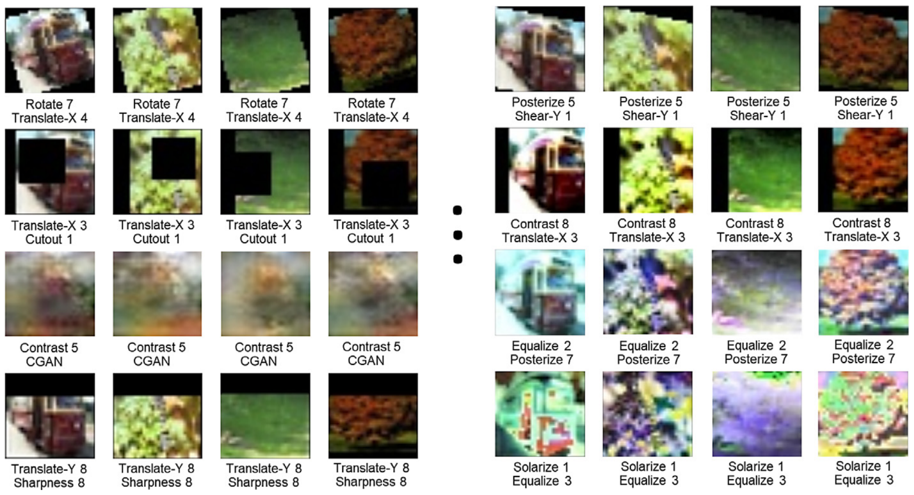

Fig. 4. Augmented images by applying each policy for a image with the controller-dependent search on the CIFAR-100 dataset.

domly. On the other hand, for the individual controller, we implement two different LSTM networks to assure the GAN and traditional operations independently. For the GAN selection controller, we use 2 cells of a single-layer LSTM only because the magnitude is not necessary to be sampled for GAN operations. It seems that dividing the policy search space into two sub-spaces which are image manipulation and generation. The comparison results of using both common and individual controllers are shown in Table 4.

3.4. Adversarial joint training

Based on adversarial training between the policy controller and target network, we jointly optimize target network training and policy search. Given a training sample $\mathbf{x}$, we denote the policy network as $\mathcal{P}(\mathbf{x}, \theta)$. The policy network aims at increasing the loss of a target network $\mathcal{T}(x, \omega)$. In specific, $\mathcal{P}(\mathbf{x}, \theta)$ struggles to increase the loss of $\mathcal{T}(x, \omega)$ by generating harder policies. On the other hand, $\mathcal{T}(x, \omega)$ attempts to minimize its target loss. For convenience, the problem of training $\mathcal{T}(\cdot, \omega)$ by minimizing the target loss $\mathcal{L} = [\mathcal{T}(a(\mathbf{x}), \omega), y]$ can be expressed with some random augmentation operation $a(\mathbf{x})$ as follows:

$$\omega^* = \text{argmin}_{\omega} \mathbb{E}_{\mathbf{x}\sim\Omega} \mathbb{L}[\mathcal{T}(a(\mathbf{x}), \omega), y] \tag{1}$$

where Ω is a training set, and each sample consists of an image $\mathbf{x}$ and class label y.

For improving the generalization and efficient training of a target network, we can use the predictions $\tau(x)$ (i.e., augmentation policies) of the policy network when augmenting an input sample $\mathbf{x}$. Therefore, the above minimization problem can be represented with $\tau(x)$ as

$$\omega^* = \text{argmin}_{\omega} \mathbb{E}_{\mathbf{x}\sim\omega} \mathbb{E}_{\tau\sim\mathcal{P}(\mathbf{x},\theta)} \mathbb{L}[\mathcal{T}(\tau(\mathbf{x}), \omega), y] \tag{2}$$

For N batches, we can represent the loss for each augmentation policy $\tau(m)$ as

$$\mathbb{L}_m = \frac{1}{N} \sum_{n=1}^{N} \mathbb{L}[\mathcal{T}(\tau_m, (\mathbf{x}_n), \omega), y_n] \tag{3}$$

This problem can be solved using the stochastic gradient descent (SGD). With α learning rate and M policies ($M \in \{2, 4, 8, 16, 32\}$ in our experiment), the training procedure of the target network is

$$\omega_{t+1} = \omega_t - \alpha \frac{1}{M{\cdot}N} \sum_{m=1}^{M} \sum_{n=1}^{N} \mathbb{L}[\mathcal{T}(\tau_m, (\mathbf{x}_n), \omega), y_n] \tag{4}$$

Using Eq. (3), we can represent the loss in brief

$$\omega_{t+1} = \omega_t - \alpha \frac{1}{M} \sum_{m=1}^{M} \omega^* \mathbb{L}_m \tag{5}$$

Training a policy controller network can be achieved by maximizing the loss Eq. (3) for each policy. Using the REINFORCE algorithm [35], we can learn the policy network. Let p_m denote the probability for the policy τ_m which is differentiable w.r.t θ of the network. The controller network can be updated with the learning rate β as

$$\theta^*_{e+1} = \theta_e + \beta \frac{1}{M} \sum_{m=1}^{M} \widehat{\mathbb{L}}_m \nabla_\theta \log \mathbf{p}_m \tag{6}$$

where $\widehat{\mathbb{L}}_i$ is the normalized moving average over the mini-batches. The moving average loss is considered as the reward of the policy network. In our implementation, we use a single-layer LSTM as an RNN controller. In a sequential manner, this controller predicts 4 actions sequentially which correspond to 2 augmentation operations and magnitudes.

4. Implementation

In this section, we present implementation details of the main modules in our GA3N shown in Fig. 1. We adopt the search space of AutoAugment for the 16 traditional augmentation operations as used in AutoAugment and Adversarial AutoAugment. To extend this search space, we train the generative GAN models (i.e., CGAN and DAGAN) for synthetic image generation. When augmenting image samples in a training batch, we use updated policies within a search space shown in Fig. 2. We sequentially apply each operation of a policy for a given image. From adversarial training between a target network (classifier) and policy controller network, the best augmentation policy can be updated at each iteration. As mentioned, the adversarial training is achieved by minimizing and maximizing a loss of a target network during alternative training. The overall algorithm for training our GA3N can be shown in Algorithm 1.

Algorithm 1: Adversarial training of GA3N.

```
Input: Samples with images $\mathbf{x}$ and object labels $y$
Output: Trained target $\mathcal{T}$ and policy $\mathcal{P}$ networks
1  Initializing parameters of $\mathcal{T}(\mathbf{x}, \omega)$ and $\mathcal{P}(\mathbf{x}, \theta)$;
2  // CGAN or DAGAN pre-training;
3  for 100 epochs do
4      for K iterations do
5          Construct a batch of J samples consisting of noise,
           image, and label triplets ;
6          Update the discriminator with the batch using SGD;
7      end
8      Construct a batch of J samples consisting of noise, image,
       and label triplets ;
9      Update the generator with the batch using SGD ;
10 end
11 // Target $\mathcal{T}$ and policy $\mathcal{P}$ network training;
12 for N epochs do
13     Initialize $\{\mathbb{L}_m\}_{m=1}^{M}$ losses for M policies;
14     Predict $\{p_m\}_{m=1}^{M}$ probabilities for M policies ;
15     for K iterations do
16         Construct a batch by applying M policies for each real
           image;
17         Evaluate $L_m$ using Eq. (3);
18         Update $\mathcal{T}$ using Eq. (5);
19     end
20     Update $\mathcal{P}$ using Eq. (6)
21 end
```

4.1. Policy controller

For making policy predictions more accurately at the current iteration, leveraging predictions at the previous iteration is crucial. To achieve this, we can use a recurrent neural network (RNN) in order to predict the current policy by feeding previous predictions as inputs. However, a simple RNN has limited short-term memory, and could not scale-up the problem of predicting policy within a large search space.

To alleviate this problem, we can use long short-term memory as the augmentation policy network as also used in [5,7]. We implement a single-layer LSTM and use it for the predictions of several operations and magnitudes as shown in Fig. 5. Here, for the sequential predictions, the parameters of the controller network are shared and fixed for efficient training. As short-term predictions, the 4 parameters for 2 operations for one sub-policy are inferred. During $M \times 5$ iterations, M policies with 5 sub-policies are predicted and propagated to the policy applier in the generative part shown in Fig. 1. The sensitivity analysis for the hyperparameter M are provided in Section 5.4.

4.2. Generative adversarial networks for data augmentation

Because the variability of augmented images by using the traditional operations only is rather limited, we introduce two GANs in order to generate new synthetic images. For improving the quality and variability of the synthetic images together, we should generate synthetic images within a class and between classes. To this end, we use conditional GAN which can generate class-wise synthetic images more accurately by feeding the condition class labels to a generator and discriminator. On the other hand, we introduce DAGAN for improving cross-class synthetic image generation. Moreover, both GANs show the effective adversarial training and good generalization ability for the slow-data regime.

In our implementation, we pre-train CGAN and DAGAN on CIFAR-100 and Tiny-ImageNet datasets. Therefore, both GANs have the basic ability to generate synthetic images on each dataset. Then, we integrate the trained GANs into GA3N and use them as additional data augmentation operations. We present more details of CGAN and DAGAN implementation in the next section.

4.2.1. Conditional GAN (CGAN)

CGAN [11] is an extended version of GAN by using an auxiliary variable **y** (e.g. image patches, class labels and attributes). The losses of generator and discriminator of this CGAN are dependent on **y** as follows:

$$\min_G \max_D \underset{x\sim p_{data}(x|y)}{\mathbb{E}}[\log D(x|y)] + \underset{x\sim p_z(z)}{\mathbb{E}}[\log(1 - D(G(z|y)))] \tag{7}$$

The auxiliary data y is fed to both generators and discriminators as inputs.

x and z represent the training image and an input noise, respectively. The overall architecture of this generator is shown in Fig. 6. We feed the generator an (RGB) colored image x of 32×32, its label y and 2-dimensional random noise z that is distributed in Gaussian distribution.

About the discriminator, we build a network that aims to distinguish the real and synthetic images. The discriminator has Dropout, Embedding layers as an Input layer, Convolution, Leaky ReLU, and Batch Normalization layers.

We then feed the real and synthetic images combined with the auxiliary y to the discriminator and use the predicted classes for computing Eq. (7). We set the dropout probability as 0.75 and the Leaky ReLU scale as 0.2. After going through convolutional, batch normalization, and LeakyReLU layers, the discriminator outputs a scalar prediction score, in the final layer. To train CGAN, we set the batch size to 96 on both CIFAR-100 and Tiny-ImageNet datasets and train for 100 epochs. As an optimizer, we use the Adam optimizer for the discriminator with 0.0002 learning rate and 0.5 beta

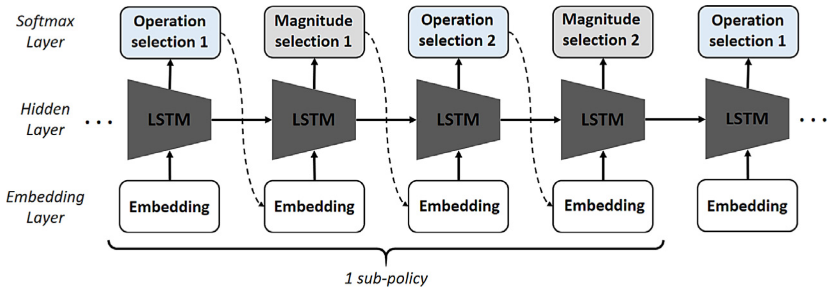

Fig. 5. The architecture of the policy controller network implemented by the shared LSTM.

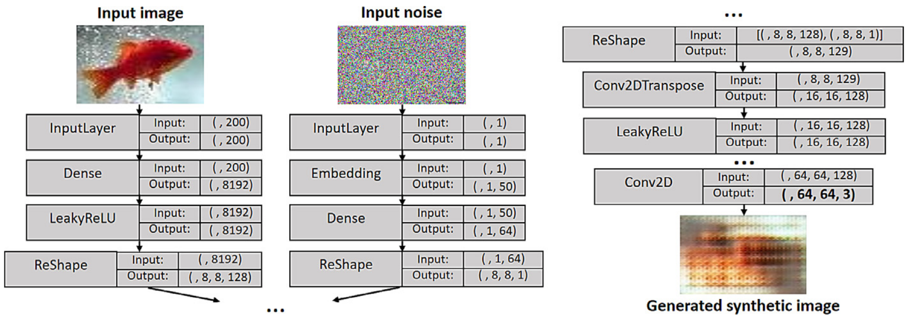

Fig. 6. The generator architecture of conditional GAN. Given the condition variable (i.e., an image), the generator produces a synthetic image with a random noise image.

value. The model parameters for the training generator are set similarly to the discriminator.

4.2.2. Data augmentation GAN (DAGAN)

DAGAN [6] has a similar architecture as CGAN in order to generate synthetic images for data augmentation. Different from CGAN, it uses a Gaussian latent vector but also embedding representation from a generator as inputs when generating a new image. With the auxiliary input, the generator of DAGAN produces more realistic and various images for within-class objects, and it can be further applied for image generations for novel classes.

Given a new image $\mathbf{x}$ from a generator, we can obtain the meaningful embedding representation with a generator $\mathbf{r}$ by $\mathbf{r} = g(\mathbf{x})$. We feed $\mathbf{r}$ and random latent vector z from the standard Gaussian distribution to an augmentative neural network $f(\cdot)$ to generate a new image x^* by using $x^* = f(r, z)$. At the next iteration, we use x^* as an input of $g(\cdot)$, and the same processes are repeated until the number of augmented samples is enough.

5. Experiments

In this section, we provide evaluation and comparison results of GA3N for object classification. We evaluate our GA3N on public available CIFAR-100, Tiny-ImageNet, and ImageNet datasets.

5.1. Datasets and backbones

We provide the details of the datasets used for evaluation. CIFAR-100 [36] contains color images of the size 32×32 for 100 object classes, and each class has 600 sample images. In total, 60K images are provided. We use 50K images and 10K images for training and validation, respectively.

ImageNet is the large-scale dataset which has over 1.2 million training, 50k validation and 10k test images for 1000 object classes, respectively.

Tiny-ImageNet dataset is the data subset of ImageNet [2]. The amount of this dataset is around 20% of ImageNet. There are 200 object classes, and 500 color images of the size 64×64 are given for each class. Also, 500 images for each class contain 50 valuation and test images.

For the backbone of our target network, we use WideResNet28-10 network on CIFAR-100. Similar to adversarial AutoAugment [7], we use the ResNet-50 on Tiny-ImageNet.

5.2. Evaluation metrics

As evaluation metrics, we compute Top-1 and Top-5 error rates which are commonly used for classification evaluation [7]. In the Top-1 score, a class prediction with the highest probability is matched with the corresponding ground truth (GT) object label or not. On the other hand, the Top-5 error rate is evaluated by the best 5 predictions (with the highest probabilities for an object class) with the object GT label. Thus, a predicted result is considered as true positive when the matched labels are the same. 100% accuracy for each metric means that all predicted results of a classifier are perfectly matched with the corresponding GT labels.

5.3. Comparison with other AutoAugment methods

To show the effects of our GA3N, we compare our method with several augmentation methods as shown in Table 2 and 3 on CIFAR-100 and Tiny-ImageNet datasets, respectively. We provide the details of the reimplementation for other AutoAugment methods and compare results in this section.

5.3.1. Reimplementation of recent AutoAugment methods

For more a fair comparison, we have re-implemented AutoAugment [5] and Adversarial AutoAugment [7]. Also, the official codes are not available to the public. We apply the same backbone and augmentation policy network for these methods. The main difference from our GA3N is that they use 16 augmentation operations excluding GAN operations.

5.3.2. Comparison results

To prove the effectiveness of our GA3N, we compare our method with several augmentation and classification methods on CIFAR-100 and Tiny-ImageNet datasets as shown in Table 2 and 3,

Table 2
Evaluation results of our GA3N by using different the number of policies (M) and policy selection methods on CIFAR-100 datasets. We also compare our GA3N with recent augmentation methods: AutoAugment [5], Adversarial AutoAugment [7], PuzzleMix [37] and Stochastic Weight Averaging (SWA) [38]. For more comparison, we re-implement AA Re-Impl [7] and evaluated with different M policies. **Bold** and italic fonts indicate the best result of our GA3N and the best results of other methods, respectively.

Methods	Number of policies (M)						
	2	4	8	16	32	N/A	Average
	Top1 / Top5	Top1 / Top5	Top1 / Top5	Top1 / Top5	Top1 / Top5	Top1 / Top5	Top1 / Top5
GA3N (Controller-Dependent)	23.4% / 9.1%	20.5% / 7.0%	15.48% / 4.4%	**15.35% / 4.1%**	19.5% / 6.7%	- / -	19.23% / 6.46%
GA3N (GAN-Dependent)	25.3% / 11.3%	22.2% / 9.7%	19.8% / 7.1%	17.3% / 6.8%	18.4% / 8.4%	- / -	20.6% / 8.66%
AutoAugment [5]	- / -	- / -	- / -	- / -	- / -	20.0% / -	- / -
Adversarial AutoAugment [7]	- / -	- / -	*15.67%* / -	- / -	- / -	- / -	- / -
AA Re-Impl [7]	25.8% / 11.6%	22.7% / 9.0%	*15.95% / 7.5%*	16.4% / 8.3%	21.2% / 9.8%	- / -	20.41% / 9.24%
PuzzleMix [37]	- / -	- / -	- / -	- / -	- / -	15.95% / -	- / -
SWA [38]	- / -	- / -	- / -	- / -	- / -	15.84% / -	- / -

Table 3
Evaluation results of our GA3N by using different the number of policies (M) and policy selection methods on Tiny-ImageNet dataset. We also compare our GA3N with recent augmentation methods: Tiny-ImageNet Challenge [12]. For more comparison, we re-implement AA Re-Impl [7] and evaluated with different M policies. **Bold** and italic fonts indicate the best result of our GA3N and the best results of other methods, respectively.

Methods	Number of policies (M)						
	2	4	8	16	32	N/A	Average
	Top1 / Top5	Top1 / Top5	Top1 / Top5	Top1 / Top5	Top1 / Top5	Top1 / Top5	Top1 / Top5
GA3N (Controller-Dependent)	60.0% / 38.1%	55.7% / 28.7%	47.6% / 25.5%	**45.2% / 23.2%**	47.2% / 25.3%	- / -	51.6% / 27.94%
GA3N (GAN-Dependent)	65.3% / 38.6%	59.7% / 30.2%	54.1% / 27.4%	49.9% / 24.1%	58.7% / 28.2%	- / -	57.54% / 29.7%
AA Re-Impl [7]	66.7% / 40.2%	61.2% / 33.9%	52.7% / 28.3%	54.6% / 29.5%	60.8% / 32.7%	- / -	59.2% / 32.92%
Vanilla model [39]	- / -	- / -	- / -	- / -	- / -	*46.3%* / -	- / -

respectively. As mentioned, the experimental results of AutoAugment [5] and Adversarial AutoAugment [7] are made based on our re-implementation. In Table 2, we have found that the AutoAugmentation is indeed beneficial for improving the classification accuracy when comparing with PuzzleMix [37] and SWA [38]. In particular, our GA3N with $M = 16$ achieves the best accuracy among them. This result is slightly better about 0.32% than the state-of-the-art result of [7]. On Tiny-ImageNet datasets, we have compared our results with the vanilla model [39] with ResNet-50 of this Tiny-ImageNet challenge as in Table 3. By using the same backbone, our GA3N with $M = 16$ produces better accuracy by 1.1% than [39].

For more comparison, we have re-implemented Adversarial AutoAugment [7] (AA Re-Impl) with the same backbones for more comparison. As shown in Table 2, the score of re-implemented AA Re-Impl (15.95%) is very close to the original score [7] (15.67%) on the CIFAR-100 set. As shown in Table 2, our GA3N achieves the better scores by 0.6%/3.4% (Top1/Top5) than scores of AA Re-Impl on CIFAR-100. In particular, the accuracy gap between the GA3N and AA Re-Impl is 7.5%/5.1% (Top1/Top5) on the Tiny-ImageNet. Therefore, these comparison results obviously show the superiority of our GA3N.

5.4. Ablation study

To evaluate each method of our GA3N, we conduct the ablation study. We first evaluate GA3N with the different number of policies (M) as shown in Table 2 and 3. We change M from 2 to 32. As M increases, the classification accuracy gets higher in general. This indicates that the diversity of augmentation operations in a policy set is important for improving a classification model. On both sets, we obtain the best results with $M = 16$. The accuracy is degraded when $M = 32$. This means that using the same augmentation several times is not effective.

In addition, we compare the GA3N with controller-dependent search and GAN-dependent search methods (refer to Section 3.3). In this comparison, we use different M. In most cases, the controller-dependent search provides higher gains. For the GAN-dependent search, we implement common and individual controllers to investigate the effect of this search. In both controllers based on GAN-dependent search, one of the GAN operations should be contained in a sub policy. In the former one, we select GAN operations randomly without using the additional controller. On the other hand, we use an extra LSTM shown in Fig. 5 for selecting GAN operations in the latter controller. Compared to the LSTM which controls 16 image manipulation operations, this one consists of 2 cells only because prediction for operation magnitude is not necessary. Table 4 shows the comparison results. Here, we also apply different M for both controllers. The common controller shows better accuracy than the individual controller. To sum up, the accuracy of each controller can be ordered as follows: controller-dependent search $\geq$ GAN-dependent search using the common controller $\geq$ GAN-dependent search using the individual controller. This also means that the diversity of augmentation operation is crucial when building a mini-batch with augmented samples. Also, using many controllers is not useful for improving the accuracy of a target network.

5.5. Proxy evaluation on CIFAR-100 and Tiny-ImageNet

In order to show the effectiveness of data augmentation, we evaluate our GA3N with proxy sets of the CIFAR-100 and Tiny-ImageNet datasets as shown in Table 5 and 6. To create the proxy training sets with a different number of training samples, we make several proxy sets consisting of 10%, 20%, and 50% of the whole training samples. To this end, we randomly select images for each class with the same ratios because the number of images for each class in both datasets is the same. But, we maintain the original validation and test sets. Once several proxy training sets are generated, we train our GA3N. When training a target network, we fix the size of the mini-batch, but change the number of training iterations per epoch in consideration of the size of proxy sets. Then, we collect a target loss (i.e. reward) for each augmentation policy and use this for the training of the policy controller.

As shown in Table 5, our GA3N achieves comparable accuracy with 50% samples with other methods [5,7,40] on the

Table 4
Comparison between the common and individual controllers by changing *M* on CIFAR-100 and Tiny-ImageNet datasets.

Dataset / Controller	CIFAR-100 / Common Controller	CIFAR-100 / Individual Controller	Tiny-ImageNet / Common Controller	Tiny-ImageNet / Individual Controller
	Top1 / Top5	Top1 / Top5	Top1 / Top5	Top1 / Top5
M=2	25.3% / 11.3%	25.8% / 13.7%	65.3% / 38.6%	66.2% / 40.3%
M=4	22.2% / 9.7%	23.1% / 10.2%	59.7% / 30.2%	59.9% / 32.5%
M=8	19.8% / 7.1%	20.4% / 8.8%	54.1% / 27.4%	54.4% / 28.5%
M=16	17.3% / 6.8%	17.7% / 7.3%	49.9% / 24.1%	51.2% / 24.6%
M=32	18.4% / 8.4%	19.0% / 9.7%	58.7% / 28.2%	59.3% / 29.1%

Table 5
Comparisons of data augmentation methods with the different number of policies *M* and the different ratios of used real training samples on the CIFAR-100 dataset. This dataset contains 50k training images. We indicate the best scores of our methods and other methods with **Bold** and italic fonts indicate on each proxy sets.

			Sample ratio used for training among whole samples			
Methods	Augmentation	Number of policies (*M*)	10% (5k images)	20% (10k images)	50% (25k images)	100% (50k images)
			Top1 / Top5	Top1 / Top5	Top1 / Top5	Top1 / Top5
GA3N	Generative	2	42.7% / 23.1%	36.0% / 19.7%	29.9% / 13.9%	23.4% / 9.1%
(proposed)	Adversarial	4	39.4% / 22.5%	33.2% / 16.8%	27.3% / 11.2%	20.5% / 7.0%
	AutoAugment	8	33.2% / 19.4%	24.8% / 11.4%	18.7% / 6.4%	15.48% / 4.4%
		16	31.9% / 16.3%	22.4% / 9.9%	18.1% / 6.2%	**15.35% / 4.1%**
		32	36.4% / 17.4%	29.1% / 11.8%	22.7% / 8.7%	19.5% / 6.7%
[5]	AutoAugment	-	- / -	- / -	- / -	17.4% / -
[7]	Adversarial AutoAugment	8	- / -	- / -	- / -	*15.67% / -*
[40]	-	-	- / -	- / -	- / -	20.43% / -

Table 6
Comparisons of data augmentation methods with the different number of policies *M* and the different ratios of used real training samples on the Tiny-ImageNet dataset. This dataset contains 50k training images. We indicate the best scores of our methods and other methods with **Bold** and Italic fonts indicate on each proxy sets.

			Sample ratio used for training among whole samples			
Methods	Augmentation	Number of policies (*M*)	10% (5k images)	20% (10k images)	50% (25k images)	100% (50k images)
			Top1 / Top5	Top1 / Top5	Top1 / Top5	Top1 / Top5
GA3N	Generative	2	76.1% / 56.2%	68.8% / 40.8%	59.1% / 36.9%	60.0% / 38.1%
(proposed)	Adversarial	4	72.5% / 51.6%	66.4% / 37.4%	54.7% / 34.4%	55.7% / 28.7%
	AutoAugment	8	67.2% / 49.3%	58.7% / 34.7%	51.8% / 27.2%	47.6% / 25.5%
		16	64.5% / 46.9%	56.9% / 32.1%	45.5% / 23.2%	**45.2% / 23.2%**
		32	66.3% / 54.1%	55.5% / 37.9%	49.1% / 28.4%	47.5% / 25.3%
Vanilla model [39]	-	-	- / -	- / -	- / -	*46.3% / -*

Table 7
Comparison of GA3N and AutoAugment methods by using different GANs and *M* on CIFAR-100 dataset.

		Sample ratio used for training among whole samples			
Methods	Number of policies (*M*)	10% (5k images)	20% (10k images)	50% (25k images)	100% (50k images)
		Top1 / Top5	Top1 / Top5	Top1 / Top5	Top1 / Top5
GA3N + CGAN	$M = 16$	32.5% / 18.9%	23.1% / 12.0%	18.7% / 7.1%	15.8% / 4.3%
GA3N + DAGAN	$M = 16$	33.7% / 19.5%	25.3% / 12.1%	20.2% / 7.3%	16.6% / 5.4%
GA3N+CGAN+DAGAN	$M = 16$	31.9% / 16.3%	22.4% / 9.9%	18.1% / 6.2%	**15.35% / 4.1%**
GA3N+CGAN+DAGAN	$M = 8$	33.2% / 19.4%	24.8% / 11.4%	18.7% / 6.4%	15.48% / 4.4%
AA [7]	$M = 8$	-/-	-/-	-/-	*15.67% / -*
AA Re-Impl [7]	$M = 8$	*36.4% / 22.1%*	*27.6% / 13.2%*	*19.5% / 8.2%*	15.95% / 7.5%
AA Re-Impl [7]	$M = 16$	38.2% / 24.7%	29.8% / 14.7%	20.6% / 9.8%	17.1% / 8.2%

CIFAR-10 dataset. Table 6 also shows that we achieve better results with 50% samples than [39]. This evaluation proves that our GA3N is an effective data augmentation method. Therefore, it is beneficial of training a target model with limited training samples.

To show the effects of the CGAN and DAGAN more, we have compared GA3N with different GANs on each datasets in Table 7 and 8. In this study, GA3N + CGAN indicates the search space extension by using Conditional GAN only; GA3N + DAGAN is the extension for only DAGAN; and GA3N + CGAN + DAGAN illustrates the extension by using both GANs (our proposed). We have also reimplemented the adversarial AutoAugment method (i.e. AA Re-Impl [7]) for more comparisons on the proxy dataset.

As shown in Table 7 and 8, our GA3N with both GANs shows the better rates than the reimplemented Adversarial AutoAugment. In addition, our GA3N with the one of them shows the comparable results with the Adversarial AutoAugment methods. Using the CGAN shows the better result than using the DAGAN. In addition, we have provided the results of our GA3N and AA Re-Impl for $M = 8$ and $M = 16$. As discussed in [7], using $M \geq 8$ degrades the accuracy for the AutoAugment. However, our GA3N shows the better results with the higher *M*. This implicitly means that the extended search space by the integration of CGAN and DAGAN allows AutoAugment methods to harness higher *M* since the diversity of the augmented samples increases further. Remarkably, the performance difference between our GA3N and AA Re-Impl is more

Table 8
Comparison of GA3N and AutoAugment methods by using different GANs and *M* on Tiny-ImageNet dataset.

Methods	Number of policies (M)	Sample ratio used for training among whole samples			
		10% (5k images)	20% (10k images)	50% (25k images)	100% (50k images)
		Top1 / Top5	Top1 / Top5	Top1 / Top5	Top1 / Top5
GA3N + CGAN	$M = 16$	64.8% / 47.3%	57.7% / 32.8%	46.2% / 24.1%	45.5% / 23.6%
GA3N + DAGAN	$M = 16$	65.1% / 49.2%	58.4% / 33.3%	46.8% / 25.2%	45.9% / 24.3%
GA3N+CGAN+DAGAN	$M = 16$	64.5% / 46.9%	56.9% / 32.1%	45.5% / 23.2%	**45.2% / 23.2%**
GA3N+CGAN+DAGAN	$M = 8$	67.2% / 49.3%	58.7% / 34.7%	51.8% / 27.2%	47.6% / 25.5%
AA Re-Impl [7]	$M = 8$	*69.3% / 54.8%*	*61.7% / 35.5%*	*57.5% / 31.0%*	*52.7% / 28.3%*
AA Re-Impl [7]	$M = 16$	72.8% / 58.2%	63.5% / 38.4%	59.5% / 33.3%	54.9% / 30.4%

Table 9
Comparison between static augmentation and our AutoAugmentation methods on CIFAR-100 and Tiny-ImageNet datasets. We compare both methods with the similar number of augmented samples.

Dataset	Method	Number of augmentations D				Number of policies M
		1	5	10	18	4
		Top1 / Top5	Top1 / Top5	Top1 / Top5	Top1 / Top5	Top1 / Top5
CIFAR-100	Static Augmentation	40.4% / 28.6%	37.2% / 25.2%	29.3% / 18.3%	25.6% / 10.5%	- / -
	GA3N	- / -	- / -	- / -	- / -	20.5% / 7.0%
Tiny-ImageNet	Static Augmentation	72.1% / 50.3%	69.2% / 44.7%	57.3% / 35.1%	53.5% / 30.4%	- / -
	GA3N	- / -	- / -	- / -	- / -	55.7% / 28.7%

distinct on the Tiny-ImageNet dataset. Therefore, we confirm that our GA3N is the more effective augmentation method for the large-scale tasks.

5.6. Augmentation comparison

Finally, we make the comparison between static augmentation and our AutoAugment methods. In static augmentation, we randomly select the D number of augmentation operations from our extended search space. For instance, 18 augmented images are generated given a single image when $n = 18$. For a more fair comparison, we compare the results of this static augmentation with our GA3N using $M = 4$[2]. As shown in Table 9, our method can reduce the Top1 and Top5 errors on the CIFAR-100 dataset by 5.1% and 2.5%, respectively. Even though the Top1 error on the Tiny-ImageNet dataset is increased by 2.2% in our GA3N, the Top5 error is reduced by 1.7%. Note that we can achieve much better results with large M as shown in Table 6. This comparison shows that selecting proper augmentation operations is important in consideration of the learning status of a target model.

6. Conclusion

In this work, we propose a generative adversarial auto augment network (GA3N) for optimal data augmentation on target tasks. Our core idea is to generate hard and new synthetic training samples for improving the robustness and generalization of the target network. To achieve this, we introduce two adversarial training approaches. We train policy and target networks in the adversarial training manner by minimizing and maximizing a target loss. In addition, we integrate two conditional GANs (CGAN and DAGAN) for new sample generations which can be considered as unseen samples. By combining two GAN operations into traditional image manipulation operations, we show that policy search space can be extended.

From the comparison with recent AutoAugment methods, we have shown that our GA3N can achieve better accuracy on CIFAR-100 and Tiny-Image Net datasets. Also, we have evaluated the effect of each method with the extensive ablation study. In particular, from the proxy evaluation, we have shown the significant effect of our GA3N. With the half of whole real samples, we have achieved comparable accuracy with recent methods. It shows that new augmentation policies for the synthetic image generation can also contribute to boosting the accuracy of the target network. Therefore, we show that the generative models can be new suitable operations for auto augmentation. Since our GA3N does not depend on the target tasks, we confirm that our GA3N is flexible and it can be applicable for various tasks such as object detection and semantic segmentation. We also believe that our work could be one of pioneer works for future auto augmentation methods.

[2] The number of augmented images for $M = 4$ is to 20 because 4 (policies) × 5 (sub-policies) = 20 (augmented images).

Declaration of Competing Interest

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

Acknowledgments

This work was supported in part by the INHA UNIVERSITY Research Grant, in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government(MSIT) (No. NRF-2021R1F1A1049447 and NRF-2022R1C1C1009208) and in part by Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (2020-0-01389, Artificial Intelligence Convergence Research Center(Inha University)).

References

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: a large-scale hierarchical image database, 2009 IEEE Conference on Computer Vision and Pattern Recognition (2009) 248–255.
[2] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, Advances in Neural Information Processing Systems 25 25 (2012) 1097–1105.
[3] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, V. Erhan Dumitru andVanhoucke, A. Rabinovich, Going deeper with convolutions, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015) 1–9.
[4] L. Wan, M. Zeiler, S. Zhang, Y.L. Cun, R. Fergus, Regularization of neural networks using dropconnect, Proceedings of the 30th International Conference on Machine Learning, PMLR (2013) 1058–1066.
[5] E.D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, Q.V. Le, Autoaugment: learning augmentation strategies from data, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2019) 113–123.

[6] A. Anthoniou, A. Storkey, H. Edward, Data augmentation generative adversarial networks, arXiv preprint arXiv:1711.04340 (2017) (2017).
[7] X. Zhang, Q. Wang, J. Zhang, Z. Zhong, Adversarial autoaugment, International Conference on Learning Representations (2020) (2020).
[8] Z. Xu, C. Qi, G. Xu, Semi-supervised attention-guided cyclegan for data augmentation on medical images, 2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM) (2019) 563–568.
[9] S. Milz, T. Rudiger, S. Suss, Aerial generation: towards realistic data augmentation using conditional gans, Proceedings of the European Conference on Computer Vision (ECCV) Workshops (2018) (2018).
[10] G. Douzas, F. Bação, Effective data generation for imbalanced learning using conditional generative adversarial networks, Expert Systems with Applications (2018)464–471.
[11] M. Mirza, S. Osindero, Conditional generative adversarial nets, arXiv preprint arXiv:1411.1784 (2014) (2014).
[12] J. Wu, Q. Zhang, G. Xu, Tiny imagenet challenge, Stanford University (2017) (2017).
[13] Z. Zhong, L. Zheng, G. Kang, S. Li, Y. Yang, Random erasing data augmentation, AAAI Conference on Artificial Intelligence 34 (07) (2020) 13001–13008.
[14] R. Takahashi, T. Matsubara, K. Uehara, Data augmentation using random image cropping and patching for deep CNNS, IEEE Trans. Circuits Syst. Video Technol. 30 (9) (2020) 2917–2931.
[15] E. Hoffer, T. Ben-Nun, I. Hubara, N. Giladi, T. Hoefler, D. Soudry, Augment your batch: improving generalization through instance repetition, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2020) 8129–8138.
[16] F. Cen, X. Zhao, W. Li, G. Wang, Deep feature augmentation for occluded image classification, Pattern Recognit 111 (2021) 107737.
[17] K. Bandara, H. Hewamalage, Y.-H. Liu, Y. Kang, C. Bergmeir, Improving the accuracy of global forecasting models using time series data augmentation, Pattern Recognit (2021) 108148.
[18] B.-B. Jia, M.-L. Zhang, Multi-dimensional classification via knn feature augmentation, Pattern Recognit 106 (2020) 107423, doi:10.1016/j.patcog.2020.107423.
[19] S. Suh, P. Lukowicz, Y.O. Lee, Discriminative feature generation for classification of imbalanced data, Pattern Recognit 122 (2022) 108302, doi:10.1016/j.patcog.2021.108302.
[20] A. Odena, C. Olah, J. Shlens, Conditional image synthesis with auxiliary classifier gan, Proceedings of Machine Learning Research (2017) 2642–2651.
[21] G. Mariani, F. Scheidegger, R. Istrate, C. Bekas, A.C.I. Malossi, Bagan: data augmentation with balancing GAN, arXiv preprint arXiv:1803.09655 (2018) (2018).
[22] B. Bozorgtabar, D. Mahapatra, J.-P. Thiran, Exprada: adversarial domain adaptation for facial expression analysis, Pattern Recognit 100 (2020) 107111, doi:10.1016/j.patcog.2019.107111.
[23] X. Zhou, J. Zhou, P. Krahenbuhl, Bottom-up object detection by grouping extreme and center points, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2019) 850–859.
[24] R. Ma, P. Tao, H. Tang, Optimizing data augmentation for semantic segmentation on small-scale dataset, Proceedings of the 2nd international conference on control and computer vision (2019) 77–81.
[25] B. Zoph, E.D. Cubuk, G. Ghiasi, T.-Y. Lin, J. Shlens, Q. V.Le, Learning data augmentation strategies for object detection, European Conference on Computer Vision (2019) 566–583.
[26] E. Smirnov, A. Melnikov, A. Oleinik, E. Ivanova, I. Kalinovskiy, E. Luckyanets, Hard example mining with auxiliary embeddings, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (2018) 37–46.
[27] C. Huang, C.C. Loy, X. Tang, Local similarity-aware deep feature embedding, Adv Neural Inf Process Syst 29 (2016) 1262–1270.
[28] H. Oh Song, Y. Xiang, S. Jegelka, S. Savarese, Deep metric learning via lifted structured feature embedding, Proceedings of the IEEE conference on computer vision and pattern recognition (2016) 4004–4012.
[29] O.M. Parkhi, A. Vedaldi, A. Zisserman, Deep face recognition, Proceedings of the British Machine Vision Conference (BMVC) (2015) 41.1–41.12.
[30] F. Schroff, D. Kalenichenko, J. Philbin, Facenet: a unified embedding for face recognition and clustering, Proceedings of the IEEE conference on computer vision and pattern recognition (2015) 815–823.
[31] B. Zoph, Q. V. Le, Neural architecture search with reinforcement learning, 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24–26, 2017, Conference Track Proceedings (2017).
[32] J. Lemley, S. Bazrafkan, P. Corcoran, Smart augmentation learning an optimal data augmentation strategy, IEEE Access (2017) 5858–5869.
[33] D. Ho, E. Liang, X. Chen, I. Stoica, P. Abbeel, Population based augmentation: efficient learning of augmentation policy schedules, Proceedings of Machine Learning Research (2019) 2731–2741.
[34] Y. Jang, T. Zhao, S. Hong, H. Lee, Adversarial defense via learning to generate diverse attacks, Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (2019) 2740–2749.
[35] R.J. Williams, Simple statistical gradient-following algorithms for connectionist reinforcement learning, Mach Learn 8 (3) (1992) 229–256.
[36] A. Krizhevsky, G. Hinton, Learning Multiple Layers of Features from Tiny Images, University of Toronto (2009).
[37] J.-H. Kim, W. Choo, H.O. Song, Puzzle mix: exploiting saliency and local statistics for optimal mixup, International Conference on Machine Learning (2020) 5275–5285.
[38] P. Izmailov, D. Podoprikhin, T. Garipov, D. Vetrov, A.G. Wilson, Averaging weights leads to wider optima and better generalization, arXiv preprint arXiv:1803.05407 (2018).
[39] L. Sun, Resnet on tiny imagenet, Stanford University (2017) (2017).
[40] S. Zagoruyko, N. Komodakis, Wide residual network, British Machine Vision Conference (2016).

VANCHINBAL CHINBAT received the BS degree in communication and network engineering from National University of Mongolia in 2018, and MS degree in Computer Vision and Machine Learning from Incheon National University, Korea in 2022. His research interests include data augmentation, data mining, generative adversarial networks, and deep meta-learning.

SEUNG-HWAN BAE received the BS degree in information and communication engineering from Chungbuk National University, in 2009 and the MS and PhD degrees in information and communications from the Gwangju Institute of Science and Technology (GIST), in 2010 and 2015, respectively. He was a senior researcher at Electronics and Telecommunications Research Institute (ETRI) in Korea from 2015 to 2017. He was an assistant professor in the Department of Computer Science and Engineering at Incheon National University, Korea from 2017 to 2020. He is currently an assistant professor in the Department of Computer Engineering at Inha University, Korea. His research interests include multi-object tracking, object detection, deep learning, feature learning, medical image analysis, generative adversarial networks, face forensic, etc.