International Journal of Computer Vision (2025) 133:929–950
https://doi.org/10.1007/s11263-024-02206-4

AROID: Improving Adversarial Robustness Through Online Instance-Wise Data Augmentation

Lin Li[1] · **Jianing Qiu**[2] · **Michael Spratling**[1,3]

Received: 15 December 2023 / Accepted: 27 July 2024 / Published online: 24 August 2024

Abstract

Deep neural networks are vulnerable to adversarial examples. Adversarial training (AT) is an effective defense against adversarial examples. However, AT is prone to overfitting which degrades robustness substantially. Recently, data augmentation (DA) was shown to be effective in mitigating robust overfitting if appropriately designed and optimized for AT. This work proposes a new method to automatically learn online, instance-wise, DA policies to improve robust generalization for AT. This is the first automated DA method specific for robustness. A novel policy learning objective, consisting of Vulnerability, Affinity and Diversity, is proposed and shown to be sufficiently effective and efficient to be practical for automatic DA generation during AT. Importantly, our method dramatically reduces the cost of policy search from the 5000 h of AutoAugment and the 412 h of IDBH to 9 h, making automated DA more practical to use for adversarial robustness. This allows our method to efficiently explore a large search space for a more effective DA policy and evolve the policy as training progresses. Empirically, our method is shown to outperform all competitive DA methods across various model architectures and datasets. Our DA policy reinforced vanilla AT to surpass several state-of-the-art AT methods regarding both accuracy and robustness. It can also be combined with those advanced AT methods to further boost robustness. Code and pre-trained models are available at: https://github.com/TreeLLi/AROID.

Keywords Adversarial robustness · Adversarial training · Data augmentation · Automated data augmentation

1 Introduction

Deep neural networks (DNNs) are well known to be vulnerable to infinitesimal yet highly malicious artificial perturbations in their input, i.e., adversarial examples (Szegedy et al., 2014). The lack of robustness cause a crisis of security and trustworthiness for applications built on DNNs and thus hinders their further deployment in real world applications especially in the critical domains like healthcare (Qiu et al., 2023). Thus far, adversarial training (AT) has been the most effective defense against adversarial attacks (Athalye et al., 2018). AT is typically formulated as a min-max optimization problem:

$$\arg\min_{\boldsymbol{\theta}} \mathbb{E}[\arg\max_{\boldsymbol{\delta}} \mathcal{L}(\boldsymbol{x}+\boldsymbol{\delta};\boldsymbol{\theta})] \tag{1}$$

where the inner maximization searches for the perturbation $\boldsymbol{\delta}$ to maximize the loss, while the outer minimization searches for the model parameters $\boldsymbol{\theta}$ to minimize the loss on the perturbed examples.

One major issue of AT is that it is prone to overfitting (Rice et al., 2020; Wong et al., 2020). Unlike in standard training (ST), overfitting in AT, a.k.a. robust overfitting (Rice et al., 2020), significantly impairs adversarial robustness. Many efforts (Li & Spratling, 2023b; Wu et al., 2020; Dong et al., 2022; Liu et al., 2023; Liu & Satoh, 2023) have been made to understand robust overfitting and mitigate its effect. One promising solution is data augmentation (DA),

Communicated by Hong Liu.

✉ Lin Li
lin.3.li@kcl.ac.uk

Jianing Qiu
jianing.qiu17@imperial.ac.uk

Michael Spratling
michael.spratling@kcl.ac.uk

[1] Department of Informatics, King's College London, Aldwych, London WC2B 4BG, UK

[2] Department of Computing, Imperial College London, London SW7 2AZ, UK

[3] Department of Behavioural and Cognitive Sciences, University of Luxembourg, 4366 Esch-Belval, Luxembourg

which is a common technique to prevent ST from overfitting. However, many studies (Rice et al., 2020; Wu et al., 2020; Gowal et al., 2021; Rebuffi et al., 2021) have revealed that advanced DA methods, originally proposed for ST, often fail to improve adversarial robustness. Therefore, DA is usually combined with other regularization techniques such as Stochastic Weight Averaging (SWA) (Rebuffi et al., 2021), Consistency regularization (Tack et al., 2022) and Separate Batch Normalization (Addepalli, Jain, and Radhakrishnan, 2022) to improve its effectiveness. However, recent work (Li & Spratling, 2023c) demonstrated that DA alone can significantly improve AT if it has strong diversity and well-balanced hardness. This suggests that ST and AT may require different DA strategies, especially in terms of hardness. It is thus necessary to design DA schemes dedicated to AT.

IDBH (Li & Spratling, 2023c) is the latest DA scheme specifically designed for AT. Despite its impressive robust performance, IDBH employs a heuristic search method to manually optimize the DA. This search process requires a complete AT for every sampled policy, which induces prohibitive computational cost and scales poorly to large datasets and models. Hence, when the computational budget is limited, the hyperparameters for IDBH might be found using a reduced search space[1] and by employing a smaller model, leading to compromised performance.

Another issue is that IDBH, in common with other conventional DA methods such as AutoAugment (Cubuk et al., 2019) and TrivialAugment (Müller & Hutter, 2021), applies the same strategy to all samples in the dataset throughout training. The distinctions between different training samples, and between the model checkpoints at different stages of training, are neglected. We hypothesize that different data samples at the same stage of training, as well as the same sample at the different stages of training, demand different DAs. Hence, we conjecture that an improvement in robustness could be realized by customizing DA for data samples and training stages.

To address the above issues, this work proposes a bi-level optimization framework (see Fig. 1) to automatically learn **A**dversarial **R**obustness by **O**nline **I**nstance-wise **D**ata-augmentation (AROID). To the best of our knowledge, **AROID is the first automated DA method specific to adversarial robustness**. AROID employs a multi-head DNN-based policy model to map a data sample to a DA policy (see Fig. 2). This DA policy is defined as a sequence of pre-defined transformations applied with strength determined by the output of the policy model. This policy model is optimized, alongside the training of the target model, towards three novel objectives to achieve a target level of hardness and diversity. DA policies, therefore, are customized for each data instance and evolve with the target network as training progresses. This in practice produces a more globally optimal DA policy and thus benefits robustness. Importantly, the proposed policy learning objectives, in contrast to the conventional ones like validation accuracy (Cubuk et al., 2019), do not reserve a subset of the training data for validation and do not rely on prohibitively expensive inner loops for training the target model to evaluate the rewards of the sampled policies. The former ensures the entire training set is available for training to avoid potential data scarcity. The latter enables policy optimization to be much more efficient and scalable so that it is more practical for AT. Compared to IDBH in particular, this allows our approach to explore a larger space of DAs. Taking an example of optimizing the DA for CIFAR10 and PRN18, AROID took 9 h using an A100 GPU, IDBH took 412 h using an A100 GPU, and AutoAugment took 5000 h using a P100 GPU (Hataya et al., 2020).

Extensive experiments show that AROID outperforms all competitive DA methods across various datasets and model architectures while being more efficient than the previous best method (IDBH). **AROID achieves state-of-the-art robustness for DA methods** on the standard benchmarks. Besides, AROID outperforms, regarding accuracy and robustness, state-of-the-art AT methods. It also complements such robust training methods and can be combined with them to improve robustness further.

2 Related Work

Robust training. To mitigate overfitting in AT, many methods other than DA, have been previously proposed. One line of works, IGR (Ross & Doshi-Velez, 2018), CURE (Moosavi-Dezfooli et al., 2019), AdvLC (Li & Spratling, 2023b), discovered a connection between adversarial vulnerability and the smoothness of input loss landscape, and promoted robustness by smoothing the input loss landscape. Meanwhile, Wu et al. (2020) and Chen et al. (2021) found that robust generalization can be improved by a flat weight loss landscape and proposed AWP and SWA, respectively, to smooth the weight loss landscape during AT. RWP (Yu et al., 2022) and SEAT (Wang & Wang, 2022) were later proposed to further refine AWP and SWA, respectively, to increase robustness. SCARL (Kuang et al., 2023) incorporated semantic information into adversarial training. IBD (Kuang et al., 2023) distilled prior knowledge from a robust pre-trained model to enhance adversarial robustness. Many works, including MART (Wang et al., 2020), LAS-AT (Jia et al., 2022), ISEAT (Li & Spratling, 2023a), considered the dif-

[1] Search space refers to the collection of all possible data augmentation policies. Each policy consists of a set of a set of sub-policies, a data augmentation method associated with a magnitude, and a probability distribution for sampling each sub-policy to apply for data augmentation (see Fig. 2 for an illustration).

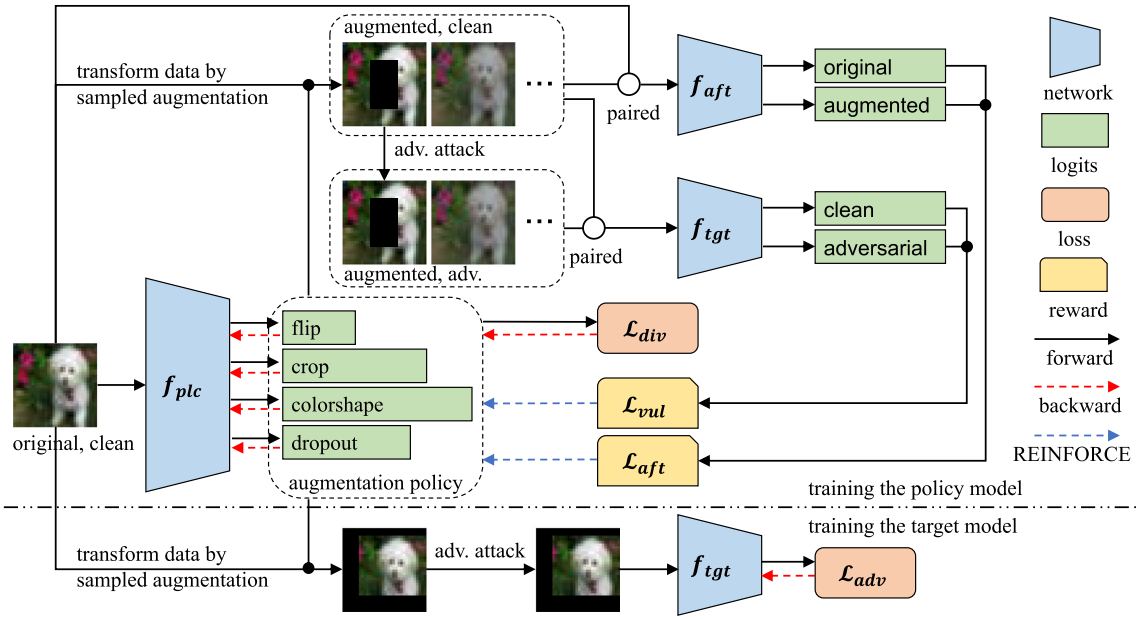

Fig. 1 An overview of the proposed method (legend in the right column). The top part shows the pipeline for training the policy model, f_{plc}, while the bottom illustrates the pipeline for training the target model, f_{tgt}. f_{aft} is a model pre-trained on clean data without any augmentation, which is used to measure the distribution shift caused by data augmentation. Please refer to Sect. 3 for a detailed explanation

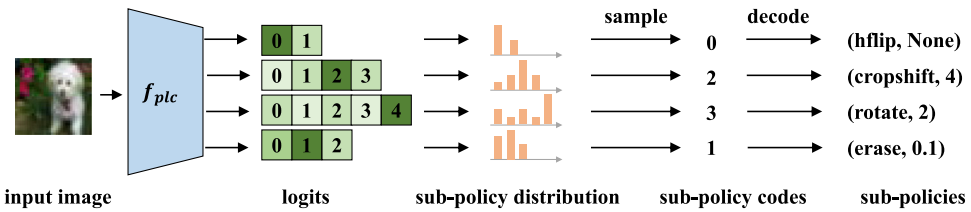

Fig. 2 An example of the proposed augmentation sampling procedure. The policy model takes an image as input and outputs logit values defining multiple, multinomial, probability distributions corresponding to different sub-policies. A sub-policy code is created by sampling from each of these distributions, and decoded into a sub-policy, i.e., a transformation and its magnitude. These transformations are applied, in sequence, to augment the image

ference between individual training instances and improved AT through regularizing in an instance-wise manner. Our proposed approach is also instance-wise, but contrary to existing methods tackles robust overfitting via DA instead of robust regularization. As shown in Sect. 4.5, it works well alone and, more importantly, complements the above techniques.

Data augmentation for ST. Although DA has been a common practice in many fields, we only review vision-based DA in this section as it is most related to our work. In computer vision, DA can be generally categorized as: basic, composite and mixup. Basic augmentations refer to a series of image transformations that can be applied independently. They mainly include crop-based (Random Crop (He et al., 2016a), Cropshift (Li & Spratling, 2023c), etc.), color-based (Brightness, Contrast, etc.), geometric-based (Rotation, Shear, etc.) and dropout-based (Cutout (DeVries & Taylor, 2017), Random Erasing (Zhong et al., 2020), etc.) transformations. Composite augmentations denote the composition of basic augmentations. Augmentations are composed into a single policy/schedule usually through two ways: interpolation (Hendrycks et al., 2020; Wang et al., 2021) and sequencing (Cubuk et al., 2019, 2020; Müller & Hutter, 2021). MixUp (Zhang et al., 2017), and analogous works like CutMix (Yun et al., 2019), can be considered as a special case of interpolation-based composition, which combines a pair of different images, instead of augmentations, as well as their labels to create a new image and its label.

Composite augmentations by design have many hyperparameters to optimize. Most previous works, as well as the pioneering AutoAugment (Cubuk et al., 2019), tackled this issue using automated machine learning (AutoML). DA policies were optimized towards maximizing validation accuracy (Cubuk et al., 2019; Lin et al., 2019; Li et al., 2020; Liu et al., 2021), maximizing training loss (Zhang et al., 2020)

or matching the distribution density between the original and augmented data (Lim et al., 2019; Hataya et al., 2020). Optimization here is particularly challenging since DA operations are usually non-differentiable. Major solutions seek to estimate the gradient of DA learning objective w.r.t. the policy generator or DA operations using, e.g., policy gradient methods (Cubuk et al., 2019; Zhang et al., 2020; Lin et al., 2019) or reparameterization trick (Li et al., 2020; Hataya et al., 2020). Alternative optimization techniques include Bayesian optimization (Lim et al., 2019) and population-based training (Ho et al., 2019). Noticeably, several works like RandAugment (Cubuk et al., 2020) and TrivialAugment (Müller & Hutter, 2021) found that if the augmentation space and schedule were appropriately designed, competitive results could be achieved using a simple hyperparameter grid search or fixed hyperparameters. This implies that in ST these advanced yet complicated methods may not be necessary. However, it remains an open question if simple search can still match these advanced optimization methods in AT. Besides, instance-wise DA strategy was also explored in Cheung and Yeung (2022); Miao et al. (2023) for ST. Our method is the first automated DA approach specific for AT. We follow the line of policy gradient methods to enable learning DA policies. A key distinction here is that our policy learning objective is designed to guide the learning of DA policies towards improved robustness for AT, while the objective of the above methods is to increase accuracy for ST.

3 Method

We propose a method to automatically learn DA alongside AT to improve robust generalization. An **instance-wise** DA policy is produced by a policy model and learned by optimizing the policy model towards three novel objectives. Updating of the policy model and the target model (the one being adversarially trained for the target task) alternates throughout training (the policy model is updated every K updates of the target model), yielding an **online** DA strategy. This online, instance-adaptive, strategy produces different augmentations for different data instances at different stages of training.

The following notation is used. $\boldsymbol{x} \in \mathbb{R}^d$ is a d-dimensional sample whose ground truth label is y. $\boldsymbol{x}_i$ refers to i-th sample in a dataset. The model is parameterized by $\boldsymbol{\theta}$. $\mathcal{L}(\boldsymbol{x}, y; \boldsymbol{\theta})$ or $\mathcal{L}(\boldsymbol{x}; \boldsymbol{\theta})$ for short denotes the predictive loss evaluated with $\boldsymbol{x}$ w.r.t. the model $\boldsymbol{\theta}$ (Cross-Entropy loss was used in all experiments). $\rho(\boldsymbol{x}; \boldsymbol{\theta})$ computes the adversarial example of $\boldsymbol{x}$ w.r.t. the model $\boldsymbol{\theta}$. $p_i(\boldsymbol{x}; \boldsymbol{\theta})$ or p_i for short refers to the output of the Softmax function applied to the final layer of the model, i.e., the probability at i-th logit given the input $\boldsymbol{x}$.

3.1 Modeling the DA Policy

Following the design of IDBH (Li & Spratling, 2023c) and TrivialAugment (Müller & Hutter, 2021), DA is implemented using four types of transformations: flip, crop, color/shape and dropout applied in order. We implement flip using HorizontalFlip, crop using Cropshift (Li & Spratling, 2023c), dropout using Erasing[2] (Zhong et al., 2020), and color/shape using a set of operations including Color, Sharpness, Brightness, Contrast, Autocontrast, Equalize, Shear (X and Y), Rotate, Translate (X and Y), Solarize and Posterize. A dummy operation, Identity, is included in each augmentation group to allow data to pass through unchanged. More details including the complete augmentation space are described in Section A.

To customize the DA applied to each data instance individually, a policy model parameterized by $\boldsymbol{\theta}_{plc}$, is used to produce a DA policy conditioned on the input data (see Fig. 2). The policy model employs a DNN backbone to extract features from the data, and multiple, parallel, linear prediction heads on the top of the extracted features to predict the policy. The policy model used in this work has four heads corresponding to the four types of DA described above. The output of a head is converted into a multinomial distribution where each logit represents a pre-defined sub-policy, i.e., an augmentation operation associated with a strength/magnitude (e.g. ShearX, 0.1). Different magnitudes of the same operation are represented by different logits, so that each has its own chance of being sampled. A particular sequence of sub-policies to apply to the input image are selected based on the probabilities encoded in the four heads of the policy network.

3.2 Objectives for Learning the Data Augmentation Policy

The policy model is trained using three novel objectives: (adversarial) Vulnerability, Affinity and Diversity. These objectives are designed to learn data augmentations with strong diversity and appropriate hardness: requirements that have been shown to be effective for adversarial training (Li & Spratling, 2023c).

3.2.1 Motivation

Intuitively, enhancing the diversity and hardness of data augmentation should help mitigate robust overfitting by increasing the complexity of the training data. Specifically, enhanced diversity increases the number of distinct data aug-

[2] Different from the original version applied at half chance, here erasing is always applied but the location and aspect ratio are randomly sampled from the given range.

mentations applied during training and expands the effective training set size (Gontijo-Lopes et al., 2021). Increasing hardness raises the difficulty level of the augmented data for the model to learn (adversarially), thereby reducing (robust) overfitting. However, if the hardness exceeds the level that the training model can fit, accuracy and even robustness will decline, despite the reduction in robust overfitting. Therefore, to maximize performance, hardness should be carefully adjusted to balance between reducing robust overfitting and improving overall performance. The optimal level of hardness should therefore be tailored to different models and training settings.

Understanding what kind of data augmentation is effective for adversarial training is not the focus of the current work so we refer the reader to (Li & Spratling, 2023c) for a formal quantitative definition of diversity and hardness, along with extensive experimental evidence supporting the above reasoning.

3.2.2 Objectives

Vulnerability measures the loss variation caused by adversarial perturbation on the augmented data w.r.t. the target model:

$$\mathcal{L}_{vul}(\boldsymbol{x};\boldsymbol{\theta}_{plc}) = \mathcal{L}(\rho(\hat{\boldsymbol{x}};\boldsymbol{\theta}_{tgt});\boldsymbol{\theta}_{tgt}) - \mathcal{L}(\hat{\boldsymbol{x}};\boldsymbol{\theta}_{tgt}) \quad \text{where } \hat{\boldsymbol{x}} = \Phi(\boldsymbol{x}; S(\boldsymbol{\theta}_{plc}(\boldsymbol{x}))) \tag{2}$$

$\Phi(\boldsymbol{x}; S(\boldsymbol{\theta}_{plc}(\boldsymbol{x})))$ augments $\boldsymbol{x}$ by $S(\boldsymbol{\theta}_{plc}(\boldsymbol{x}))$, the augmentations sampled from the output distribution of policy model conditioned on $\boldsymbol{x}$, so $\hat{\boldsymbol{x}}$ is the augmented data. A larger Vulnerability indicates that $\boldsymbol{x}$ becomes more vulnerable to adversarial attack after DA. A common belief about the relationship between training data and robustness is that AT benefits from adversarially hard samples.[3] (Madry et al., 2018; Li & Spratling, 2023c). From a geometric perspective, maximizing Vulnerability encourages the policy model to project data into the previously less-robustified space.

[3] "Adversarially hard samples" refer to samples that are difficult to classify correctly after being adversarially perturbed. The difficulty, or hardness, generally increases with the adversarial vulnerability of the original sample and the strength of the adversarial attack. From the perspective of attack strength, adversarially hard samples are those perturbed by stronger attacks. The statement "AT benefits from adversarially hard samples" can, therefore, be understood more broadly as meaning that training with stronger attacks will lead to more effective adversarial training and thus higher robustness. For example, multi-step AT is generally considered more effective than single-step AT (Madry et al., 2018) From the perspective of adversarial vulnerability, adversarially hard samples are those with higher vulnerability to attacks. Hard data augmentation can make data more susceptible to attacks, thereby producing adversarially hard samples. Empirical evidence (Li & Spratling, 2023c) suggests that adversarial training benefits from increasing the hardness of data augmentation within an appropriate range, as this helps mitigate robust overfitting and enhance performance.

Nevertheless, the maximization of Vulnerability, if not constrained, would likely favor those augmentations producing samples far away from the original distribution. Training with such augmentations was observed to degrade accuracy and even robustness when accuracy is overly reduced (Li & Spratling, 2023c). Therefore, Vulnerability should be maximized while the distribution shift caused by augmentation is constrained:

$$arg\max_{\boldsymbol{\theta}_{plc}} \mathcal{L}_{vul}(\boldsymbol{x};\boldsymbol{\theta}_{plc}) \text{ s.t. } ds(\boldsymbol{x},\hat{\boldsymbol{x}}) \leq D \tag{3}$$

where $ds(\cdot)$ measures the distribution shift between two samples and D is a constant. Directly solving Eq. (3) is intractable, so we convert it into an unconstrained optimization problem by adding a penalty on the distribution shift as:

$$arg\max_{\boldsymbol{\theta}_{plc}} \mathcal{L}_{vul}(\boldsymbol{x};\boldsymbol{\theta}_{plc}) - \lambda \cdot ds(\boldsymbol{x},\hat{\boldsymbol{x}}) \tag{4}$$

where λ is a hyperparameter and a larger λ corresponds to a tighter constraint on distribution shift, i.e., smaller D. Distribution shift is measured using a variant of the Affinity metric (Gontijo-Lopes et al., 2021):

$$ds(\boldsymbol{x},\hat{\boldsymbol{x}}) = \mathcal{L}_{aft}(\boldsymbol{x};\boldsymbol{\theta}_{plc}) = \mathcal{L}(\hat{\boldsymbol{x}};\boldsymbol{\theta}_{aft}) - \mathcal{L}(\boldsymbol{x};\boldsymbol{\theta}_{aft}) \tag{5}$$

Affinity captures the loss variation caused by DA w.r.t. a model $\boldsymbol{\theta}_{aft}$ (called the affinity model): a model pre-trained on the original data (i.e., without any data augmentation). Affinity increases as the augmentation proposed by the policy network makes data harder for the affinity model to correctly classify. By substituting Eq. (5) into Eq. (4), we obtain an adjustable Hardness objective:

$$\mathcal{L}_{hrd}(\boldsymbol{x};\boldsymbol{\theta}_{plc}) = \mathcal{L}_{vul}(\boldsymbol{x};\boldsymbol{\theta}_{plc}) - \lambda \cdot \mathcal{L}_{aft}(\boldsymbol{x};\boldsymbol{\theta}_{plc}) \tag{6}$$

This encourages the DA produced by the policy model to be at a level of hardness defined by λ (larger values of λ corresponding to lower hardness). Ideally, λ should be tuned to ensure the distribution shift caused by DA is sufficient to benefit robustness while not being so severe as to harm accuracy.

Last, we introduce a Diversity objective to promote diverse DA. Diversity enforces a relaxed uniform distribution prior over the logits of the policy model, i.e., the output augmentation distribution:

$$\mathcal{L}^h_{div}(\boldsymbol{x}) = \frac{1}{C}\left[-\sum_{i}^{p^h_i<l}\log(p^h_i) + \sum_{j}^{p^h_j>u}\log(p^h_j)\right] \tag{7}$$

C is the total count of logits violating either lower (l), or upper (u) limits and h is the index of the prediction head.

Intuitively speaking, the Diversity loss penalizes overly small and large probabilities, helping to constrain the distribution to lie in a pre-defined range (l, u). As l and u approach the mean probability, the enforced prior becomes closer to a uniform distribution, which corresponds to a highly diverse DA policy. Diversity encourages the policy model to avoid the over-exploitation of certain augmentations and to explore other candidate augmentations. Note that Diversity is applied to the color/shape head in a hierarchical way: type-wise and strength-wise inside each type of augmentation.

Combining the above three objectives together, the policy model is trained to optimize:

$$arg\min_{\boldsymbol{\theta}_{plc}} \; -\mathbb{E}_{i\in B}\mathcal{L}_{hrd}(\boldsymbol{x}_i) + \beta \cdot \mathbb{E}_{h\in H}\mathcal{L}^h_{div}(\boldsymbol{x}; \boldsymbol{\theta}_{plc}) \tag{8}$$

where B is the batch size and β trades-off hardness against diversity. $\mathcal{L}^h_{div}$ is calculated across instances in a batch, so no need for averaging over B like $\mathcal{L}_{hrd}$.

3.2.3 Mechanism

The Vulnerability objective is computed using feedback on adversarial vulnerability, measured by the variation in loss caused by adversarial perturbations, from the target model. The policy model learns from this feedback to determine which types and magnitudes of data augmentation (DA) elevates the adversarial vulnerability of augmented data. This learning raises the likelihood of applying such augmentations to the training data, thereby resulting in increased hardness. Meanwhile, the Affinity objective is employed to limit DA's hardness to a level that does not compromise performance. Additionally, the Diversity objective prevents the over-reliance on specific DA methods, promoting exploration across a diverse spectrum of augmentation techniques. Together, these three objectives dictate the appropriate DA for each training sample.

3.3 Optimization

The entire training is a bi-level optimization process (Algorithm 1): the target and policy models are updated alternately. This online training strategy adapts the policy model to the varying demands for DA from the target model at the different stages of training. The target model is optimized using AT with the augmentation sampled from the policy model:

$$arg\min_{\boldsymbol{\theta}_{tgt}} \mathcal{L}(\rho(\Phi(\boldsymbol{x}; S(\boldsymbol{\theta}_{plc}(\boldsymbol{x}))); \boldsymbol{\theta}_{tgt}); \boldsymbol{\theta}_{tgt}) \tag{9}$$

After every K updates of the target model, the policy model is updated using the gradients of the policy learning loss as follows:

$$\frac{(8)}{\partial \boldsymbol{\theta}_{plc}} = -\frac{\partial \mathbb{E}_{i\in B}\mathcal{L}_{hrd}(\boldsymbol{x}_i)}{\partial \boldsymbol{\theta}_{plc}} + \beta \frac{\mathbb{E}_{h\in H}\mathcal{L}^h_{div}(\boldsymbol{x})}{\partial \boldsymbol{\theta}_{plc}} \tag{10}$$

The latter can be derived directly, while the former $\frac{\partial \mathcal{L}_{hrd}}{\partial \boldsymbol{\theta}_{plc}}$ cannot because the involved augmentation operations are non-differentiable. To estimate these gradients, we apply the REINFORCE algorithm (Williams, 1992) with baseline trick to reduce the variance of gradient estimation. It first samples T augmentations, named trajectories, in parallel from the policy model and then computes the real Hardness value, $\mathcal{L}^{(t)}_{hrd}$, using Eq. (6) independently on each trajectory t. The gradients are estimated (see Section B for derivation) as follows:

$$\frac{1}{B \cdot T}\sum_{i=1}^{B}\sum_{t=1}^{T}\sum_{h=1}^{H}\frac{\partial \log(p^h_{(t)}(\boldsymbol{x}_i))}{\partial \boldsymbol{\theta}_{plc}}[\mathcal{L}^{(t)}_{hrd}(\boldsymbol{x}_i) - \tilde{\mathcal{L}_{hrd}}] \tag{11}$$

$p^h_{(t)}$ is the probability of the sampled sub-policy at the h-th head and $\tilde{\mathcal{L}_{hrd}} = \frac{1}{T}\sum_{t=1}^{T}\mathcal{L}^{(t)}_{hrd}(\boldsymbol{x}_i)$ is the mean $\mathcal{L}_{hrd}$ (the baseline used in the baseline trick) averaged over the trajectories. Algorithm 2 illustrates one iteration of updating the policy model. Note that, when one model is being updated, backpropagation is blocked through the other. The affinity model, used in calculating the Affinity metric, is fixed throughout training.

Algorithm 1. High-level training procedures of the proposed method. α is the learning rate. M is the number of iterations.

```
for i = 1 to M do
    // for every K iterations
    if i %K == 0 then
        // update the policy model by Algo. 2
    end
    // the policy distribution
    d = θ_plc(x_i)
    // sample & apply augmentations
    x̂_i = Φ(x_i; S(d))
    L = L(ρ(x̂_i; θ_tgt); θ_tgt)
    // update the target model
    θ_tgt = θ_tgt − α_tgt · ∇_{θ_tgt} L
end
```

3.4 Modes of Application

AROID can be used in two modes: online and offline. In the online mode, the policy and target models are jointly trained so that the policy model has to be retrained every time a new target model is trained. This adapts the DA policy to the target model on-the-fly which improves effectiveness but adds the extra cost of policy learning to that of adversarial

Algorithm 2. Pseudo code of training the policy model for one iteration. $\boldsymbol{x}$ is randomly sampled from the entire dataset.

$d = \boldsymbol{\theta}_{plc}(\boldsymbol{x})$
// same $\boldsymbol{x}$ used by all traj.
for $t = 1$ **to** T **do**
  $\hat{\boldsymbol{x}}_{(t)} = \Phi(\boldsymbol{x}, S(d))$
  $\mathcal{P}_{(t)} = \prod_{h=1}^{H} p^h_{(t)}$ // prob of traj t
  $\mathcal{L}^{(t)}_{hrd}$ // computed by Eq. (6)
end
$\tilde{\mathcal{L}_{hrd}} = \frac{1}{T}\sum_{t=1}^{T} \mathcal{L}^{(t)}_{hrd}$ // mean $\mathcal{L}^{(t)}_{hrd}$
$L = \frac{1}{T}\sum_{t=1}^{T} \log(\mathcal{P}_{(t)})[\mathcal{L}^{(t)}_{hrd} - \tilde{\mathcal{L}_{hrd}}]$
$\mathcal{L}^{(h)}_{div}$ // computed using Eq. (7)
$L = -L + \beta \frac{1}{H}\sum_{h=1}^{H} \mathcal{L}^{(h)}_{div}$
$\boldsymbol{\theta}_{plc} = \boldsymbol{\theta}_{plc} - \alpha_{plc} \cdot \nabla_{\boldsymbol{\theta}_{plc}} L$

training. In the offline mode, the training of policy and target models are separate phases. A policy model is trained in advance (using online AROID), a step that is analogous to the hyperparameter optimization of other DA methods. This pre-trained policy model is then subsequently used to train a new target model. Specifically, at each epoch of training the target network a policy network checkpoint, saved at the corresponding epoch when using online AROID, is used to sample DA policies for training the target model. When AROID is deployed in this offline mode, we refer to it as AROID-T, as it involves the transfer of the policy model. The standard mode of application is online, which we refer to simply as AROID.

3.5 Efficiency

The efficiency of AROID is dependent on the mode. The cost of AROID is composed of two parts: policy learning and DA sampling. Policy learning can be one-time expense if AROID is used in offline mode. DA sampling requires only one forward pass of the policy model, which can be negligible because the policy model can be much smaller than the target model without hurting the performance. Therefore, AROID in offline mode is roughly as efficient as other regular DA methods.

In online mode, in the worst case, AROID adds about 43.6% extra computation to baseline AT (see calculation in Section C) when $T = 8$ and $K = 5$. This is less than the overhead 52.5% of the state-of-the-art AT method LAS-AT (Jia et al., 2022) and substantially less than the search cost of IDBH and AutoAugment (compared in Sect. 4.4). Furthermore, we observed that AROID can still achieve robustness higher than other competitors with a much smaller policy model (Sect. 4.13.3), reduced T and increased K (Sect. 4.4) for improved efficiency. For example, setting $T = 4$ and $K = 20$, the overhead is only about 10% compared to baseline AT.

Another efficiency concern, as for all other deep learning methods, is hyperparameter optimization. We discuss below how this can be done efficiently so that AROID can be easily adapted to a new setting. First, as shown in Sect. 4.13.1, most of our hyperparameters can transfer well among different training settings, so that only a light tuning is needed to achieve reasonably good performance for new setting. In most cases, only λ needs to be tuned. Second, hyperparameter optimization can be accelerated by first searching with a cheap setting, such as $K = 20$ and $T = 4$, and then transferring the found values to the final setting, i.e., $K = 5$ and $T = 8$. Note that our hyperparameter tuning process is not different from others.

4 Experiments

The experiments in this section were based on the following setup unless otherwise specified.

General set-ups. We used model architectures Vision Transformer (ViT-B/16 and ViT-B/4) (Dosovitskiy et al., 2020), WideResNet34-10 (WRN34-10) (Zagoruyko & Komodakis 2016) and PreAct ResNet-18 (PRN18) (He et al., 2016b). We evaluated on datasets CIFAR10/100 (Krizhevsky, 2009), Imagenette[4] and ImageNet (Deng et al., 2009).

For CIFAR10/100, models were trained by stochastic gradient descent (SGD) for 200 epochs with an initial learning rate 0.1 divided by 10 at 50% and 75% of epochs. The momentum was 0.9, the weight decay was 5e-4 and the batch size was 128. The experiments on Imagenette and ImageNet followed a similar protocol as those on CIFAR10 except the following changes. For Imagenette, the weight decay was 1e-4, the total number of epochs was 40, and the learning rate was decayed at 36th and 38th epoch. The ViT-B/16 was pre-trained on ImageNet-1K. Gradient clipping was applied throughout training. Note that CIFAR10 with ViT-B/4 is trained using the same setting as Imagenette with ViT-B/16. For ImageNet, models were trained for 50 epochs with an initial learning rate 0.01 divided by 10 at 20th and 40th epoch. Models were pre-trained on ImageNet-1K. The weight decay was 0. Experiments were run on Nvidia Tesla V100 and A100. All results reported by us were averaged over 3 runs except for ImageNet due to the limit of computational resource.

Adversarial set-ups. By default, we used ℓ_∞ PGD AT (Madry et al., 2018) with a perturbation budget, ϵ, of 8/255. The number of steps was 10 and the step size was 2/255. For ImageNet, the perturbation budget, ϵ, was 4/255, the number of steps was 2 and the step size was $2\epsilon/3$. Following

[4] Imagenette is a subset of ImageNet consisting of 10 classes. We adopt a previous version (v1), https://s3.amazonaws.com/fast-ai-imageclas/imagenette.tgz, as suggested by Mo et al. (2022).

Table 1 The performance of various DA methods

DA method	CIFAR10				CIFAR100				Imagenette	
	WRN34-10		ViT-B/4		WRN34-10		PRN18		ViT-B/16	
	Acc.	Rob.	Acc.	Rob.	Acc.	Rob.	Acc.	Rob.	Acc.	Rob.
RandomCrop	85.83	52.26	83.04	46.72	61.44	27.98	55.04	24.83	92.73	66.47
Cutout	86.95	52.89	83.61	48.67	59.04	27.51	57.37	24.51	93.27	67.20
CutMix	86.88	53.38	80.83	47.24	58.57	27.49	57.32	25.54	93.87	70.20
AutoAugment	87.71	54.60	81.96	47.47	64.10	29.08	58.51	25.28	95.13	67.60
TrivialAugment	87.35	53.86	80.55	46.39	62.55	28.97	57.24	24.82	**95.25**	69.00
IDBH	88.61	55.29	85.09	49.63	60.93	29.03	59.38	26.24	95.20	69.93
AROID (ours)	**88.99**	**55.91**	**87.34**	**51.25**	**64.44**	**29.75**	**60.17**	**26.56**	94.88	**71.32**

Bold value indicates the best performance, and Underline value indicates the second best performance
RandomCrop is the baseline DA consists of horizontal flip and random crop with 4 padding

Rice et al. (2020), we tracked PGD10 robustness on the test set at the end of each epoch during training and selected the checkpoint with the highest PGD10 robustness, i.e., the "best" checkpoint to report robustness. Robustness was evaluated by AutoAttack (Croce & Hein, 2020).

Configuration of AROID. Hyperparameters are optimized using grid search. By default, $T = 5$, $K = 8$ and $\beta = 0.8$ were used. The diversity limits l and u were 0.9 (0.8)[5] and 4.0 respectively for CNNs (ViTs). λ was 0.4-0.2-0.1 (decayed with the learning rate for better performance), 0.4 and 0.3 for WRN34-10, ViT-B/4 and PRN18 on CIFAR10, 0.3-0.1-0.01 and 0.2 for WRN34-10 and PRN18 on CIFAR100, and 0.3 for ViT-B/16 on Imagenette. The default backbone of the policy model was PRN18 except that ViT-B/16 (pre-trained on ImageNet-1K) was used for Imagenette.[6]

Section D describes more implementation details of AROID and the competitive methods to be compared below.

4.1 Benchmarking DA on Adversarial Robustness

Table 1 compares our proposed method against existing DA methods. **AROID outperforms all existing methods regarding robustness across all five tested settings**. The improvement over the previous best method is particularly significant for ViT-B on CIFAR10 (+1.62%) and Imagenette (+1.12%). Note that in most cases IDBH is the only method whose robustness is close to ours. However, our method is much more efficient than IDBH in terms of policy search (shown in Sect. 4.4). If our method is compared only to those methods with a computational cost the same or less than AROID's, i.e., excluding IDBH and AutoAugment, the improvement over the second best method is +2.05%/2.58%/0.78%/1.12%/1.02% for the five experiments. Furthermore, we highlight the substantial improvement over the baseline of our method, +3.65%/4.53%/1.77%/4.85%/1.73%, in these five settings.

In addition, **AROID also achieves the highest accuracy in four of the five tested settings**, and in the setting of Imagenette the accuracy gap between the best method and ours is marginal (0.37%). Overall, our method significantly improves both accuracy and robustness, achieving a much better trade-off between accuracy and robustness. The consistent superior performance of our method, across various datasets (low and high resolution, simple and complex) and model architectures (CNNs and ViTs, small and large capacity), suggests that it has a good generalization ability.

4.2 Offline Versus Online AROID

This section evaluates the transferability of the learned policy models. It uses AROID in the offline mode (i.e. AROID-T as described in Sect. 3.4), across three scenarios: (1) with the same dataset and model architecture; (2) across different datasets; (3) across different model architectures. In scenario 1, a policy model is pre-trained on CIFAR10 for a WRN34-10 model and is applied to train a WRN34-10 model on CIFAR10. In scenario 2, a policy model is pre-trained on CIFAR10 for a WRN34-10 model and is applied to train a WRN34-10 model on CIFAR100. In scenario 3, a policy model is pre-trained on CIFAR10 for a PRN18 model and is applied to train a ViT-B/4 model on CIFAR10.

As shown in Table 2, AROID-T achieved accuracy and robustness comparable to its online counterpart, AROID. Importantly, AROID-T still outperforms previous data augmentation methods (Table 1) in terms of both accuracy and robustness. Notably, the cost of applying AROID-T is

[5] The value of l and u is a factor relative to the arithmetic mean chance, $\tilde{p}$, of sampling an augmentation in each group (prediction head), so the real absolute threshold value will be, e.g., $l \cdot \tilde{p}$. Taking an example of the Crop prediction head with 16 (1+15) magnitudes in total, $\tilde{p} = 1/16$.

[6] it was observed to be difficult for PRN18 to quickly fit Imagenette data to a reasonable degree in ST. Note that this ability is especially important when training on Imagenette because the total number of epochs (40) is much less than for the other datasets (200).

Table 2 The performance of AROID-T, our method in offline mode

Policy source	CIFAR10→CIFAR10		CIFAR10→CIFAR10		CIFAR10→CIFAR100	
	WRN34-10→WRN34-10		PRN18→WRN34-10		WRN34-10→WRN34-10	
	Acc.	Rob.	Acc.	Rob.	Acc.	Rob.
AROID-T	88.76	55.61	86.17	50.70	**64.97**	29.67
AROID	**88.99**	**55.91**	**87.34**	**51.25**	64.44	**29.75**

Bold value indicates the best performance
Results compare the different settings of transferring pre-trained policy models with results obtained using AROID in online mode when trained in the transfer destination setting

Table 3 Evaluation of robust overfitting for models trained with various data augmentation methods on CIFAR10/100 with WRN34-10

DA Method	CIFAR10						CIFAR100					
	Accuracy (%)			Robustness (%)			Accuracy (%)			Robustness (%)		
	Best	End	Diff.	Best	End	Diff.	Best	End	Diff.	Best	End	Diff.
baseline	85.8	86.2	**−0.3**	52.2	46.6	5.6	61.4	59.7	1.7	27.9	24.2	3.6
Cutout	86.9	87.4	−0.4	52.8	51.0	1.8	59.0	61.3	−2.2	27.5	25.0	2.4
CutMix	86.8	87.5	−0.6	53.3	49.8	3.5	58.5	62.8	−4.3	27.4	26.2	1.2
AutoAugment	87.7	88.7	−1.0	54.6	54.0	**0.5**	64.1	64.6	**−0.5**	29.0	27.1	1.9
TrivialAugment	87.3	87.7	−0.4	53.8	53.1	0.6	62.5	64.2	−1.6	28.9	27.3	1.6
IDBH	88.6	88.9	**−0.3**	55.2	53.4	1.8	60.9	64.4	−3.5	29.0	26.2	2.8
AROID (ours)	**88.9**	**89.2**	**−0.3**	**55.9**	**55.0**	0.9	**64.4**	**65.9**	−1.5	**29.7**	**28.9**	**0.8**

Bold value indicates the best performance, and Underline value indicates the second best performance

roughly the same as that of other data augmentation methods. Overall, these results demonstrate that AROID-T transfers well across various settings.

4.3 Mitigating Robust Overfitting

This section evaluates the effectiveness of our proposed method in mitigating robust overfitting. Robust overfitting is measured, using the standard convention, as the difference between the best and end robustness. The results in Table 3 demonstrate that compared to the baseline, AROID substantially reduces the degree of robust overfitting from 5.64 to 0.91% on CIFAR10 and from 3.69 to 0.83% on CIFAR100. AROID achieves the smallest robustness gap among all competitive methods on CIFAR100. Additionally, AROID achieves a robustness gap of 0.91%, close to the minimum record of 0.52% achieved by AutoAugment, while exhibiting significantly higher best and end robustness rates of +1.31% and +0.92%, respectively. Overall, these results suggest that **our method effectively mitigates robust overfitting.**

4.4 Comparison of Policy Search Costs

We compare here the cost of policy search of AROID against other automated DA methods, i.e., AutoAugment and IDBH. Before comparison, it is important to be aware that the search cost for IDBH increases linearly with the size of search space, while the cost of AROID stays approximately constant. IDBH thus uses a reduced search space that is much smaller than the search space of AROID. However, reducing the search space depends on prior knowledge about the training datasets, which may not generalize to other datasets. Moreover, scaling IDBH to our larger search space is intractable, and it would be even more intractable if IDBH was applied to find DAs for each data instance at each stage of training, as is done by AROID.

Even in the most expensive configuration ($K = 5$ and $T = 8$), AROID is substantially cheaper than IDBH and AutoAugment regarding the cost of policy search as shown in Table 4. The computational efficiency of AROID can be further increased by reducing the policy update frequency (increasing K) and/or decreasing the number of trajectories T, while still matching the robustness of IDBH. If IDBH and AutoAugment were restricted to use the same, much lower, budget for searching for a DA policy, given the huge gap, we suspect that they may find nothing useful.

4.5 Comparison with State-of-the-Art Robust Training Methods

Table 5 compares our method against state-of-the-art robust training methods. It can be seen that AROID substantially improves vanilla AT in terms of accuracy (by 3.16%) and robustness (by 3.65%). This improvement is sufficient to boost the performance of vanilla AT to surpass the state-of-the-art robust training methods like SEAT and LAS-AWP in terms of both accuracy and robustness. This suggests that

Table 4 The cost of policy search for automated DA methods using PRN18 on CIFAR10

Method	K	T	Acc.	Rob.	Search space				Time
					Prior	Probability	Magnitude	Size	
AutoAugment	–	–	83.27	49.20	No	Discrete	Discrete	2.9×10^{32}	5000
IDBH	–	–	84.23	50.47	Yes	Discrete	Discrete	80	412.83
AROID	5	8	**84.68**	**50.57**	No	Continuous	Discrete	Uncountable	9.51
AROID	20	8	84.11	50.45	No	Continuous	Discrete	Uncountable	6.85
AROID	20	4	83.63	50.52	No	Continuous	Discrete	Uncountable	**6.24**

Bold value indicates the best performance, and Underline value indicates the second best performance
AROID is used in online mode. The size of search space counts the possible combinations of probabilities and magnitudes. Our search space is uncountable due to its continuous range of probability, and is much larger than that of IDBH as it covers a much wider range of probabilities and magnitudes. Time denotes the total hours required for one search over the search space using an Nvidia A100 GPU for IDBH and AROID and a P100 GPU for AutoAugment (data is copied from Hataya et al. (2020))

Table 5 The performance of various robust training (RT) methods with baseline and our augmentations for WRN34-10 on CIFAR10

RT method	DA method	Acc.	Rob.
AT	RandomCrop	85.83	52.26
AT-SWA	RandomCrop	84.30	54.29
AT-AWP	RandomCrop	85.93	54.34
AT-RWP	RandomCrop	86.86	54.61
MART	RandomCrop	84.17	51.10
MART-AWP	RandomCrop	84.43	54.23
SEAT	RandomCrop	86.44	55.67
LAS-AT	RandomCrop	86.23	53.58
LAS-AWP	RandomCrop	87.74	55.52
AT-SWA	CutMix	87.65	56.03
AT	AROID (ours)	**88.99**	55.91
AT-SWA	AROID (ours)	87.84	56.67
AT-AWP	AROID (ours)	87.94	56.98
AT-AWP-SWA	AROID (ours)	88.39	**57.03**

Bold value indicates the best performance, and Underline value indicates the second best performance

Table 6 Comparison of various DA methods when trained by alternative AT methods like TRADES and SCORE for PRN18 on CIFAR10

DA Method	TRADES		SCORE	
	Acc.	Rob.	Acc.	Rob.
RandomCrop	83.01	49.10	80.15	48.88
Cutout	81.74	48.98	82.02	50.08
AutoAugment	80.76	48.64	81.68	49.93
TrivialAugment	80.91	48.04	80.39	49.24
IDBH	82.24	50.86	82.35	50.97
AROID (ours)	**84.04**	**51.33**	**82.69**	**51.18**

Bold value indicates the best performance, and Underline value indicates the second best performance
λ is 0.6 for TRADES and 0.3 for SCORE. The other hyperparameters are configured by default as specified in Section D

our method achieved a better trade-off between accuracy and robustness while boosting robustness.

More importantly, our method, as it is based on DA, can be easily integrated into the pipeline of existing robust training methods and, as our results show, is complementary to them. Our method was combined with other AT methods in the same way as any other data augmentation method: simply by using the sampled data augmentation policy to augment the data before generating adversarial examples. The update of the policy model is independent of the training method used. By combining with SWA and/or AWP, our method substantially improves robustness even further while still maintaining an accuracy higher than that achieved by others methods. It is worth noting that CutMix combined with SWA is widely recognized as a strong baseline for data augmentation. Our approach surpasses this baseline when combined with SWA as well.

4.6 Generalization to Alternative AT Methods

To further test the generalizability of AROID to alternative AT methods, we integrate AROID with two more superior AT methods: TRADES (Zhang et al., 2019) and SCORE (Pang et al., 2022). Results are shown in Table 6. AROID achieves highest accuracy and robustness among all the tested DA methods with both advanced AT methods. Overall, these results together with those in Sect. 4.5, show that AROID generalizes well to various AT methods (PGD, TRADES, SCORE, AWP, SWA).

4.7 Combining with Extra Data

The leading methods on the robustness benchmark RobustBench (Croce et al., 2021) heavily use extra data to augment adversarial training. We incorporate AROID with extra real data following Carmon et al. (2019) and compare it against PORT (Sehwag et al., 2022) and HAT (Rade & Moosavi-Dezfooli, 2022) which are ranked, to date, first and second respectively in RobustBench for the model architecture WRN34-10. As shown in Table 7, our method significantly

Table 7 The performance of our methods when trained with extra data for WRN34-10 on CIFAR10

Method	AT	Activation	Extra data	DA	Epochs	Batch	Acc.	Rob.
baseline	PGD10	ReLU	0.5M Real	RandomCrop	200	128	88.78	57.95
PORT	PGD10	ReLU	10 M Synthetic	RandomCrop	200	128	86.68	60.27
ours	PGD10	ReLU	0.5M Real	AROID	200	128	<u>92.38</u>	61.49
baseline	PGD10	ReLU	0.5M Real	RandomCrop	400	512	89.66	58.73
HAT	HAT	SiLU	0.5M Real	RandomCrop	400	512	91.47	62.83
ours	PGD10	ReLU	0.5M Real	AROID	400	512	**92.48**	62.60
BDM	PGD10	ReLU	50 M Synthetic	RandomCrop	400	512	92.06	<u>63.39</u>
ours	PGD10	ReLU	50 M Synthetic	AROID	400	512	92.28	**63.56**

Bold value indicates the best performance, and Underline value indicates the second best performance
All compared methods use 0.5M extra real data except for PORT and BDM which use 10M and 50M synthetic data, respectively. "Activation" refers to the activation function in the model architecture. "Batch" denotes the batch size. The results of PORT and HAT are copied from RobustBench

improves both accuracy and robustness over the baseline methods. Our method also surpasses PORT regarding both accuracy and robustness. Our method, compared to HAT, achieves a comparable robustness and a clearly higher accuracy exhibiting a better trade-off between accuracy and robustness. Note that HAT employs a more effective AT method, HAT, and a different activation function, SiLU, both of which are known to boost performance.

Next, we test whether AROID can be applied to enhance the state-of-the-art method BDM (Wang et al., 2023), which utilizes 50 M synthetic data samples. As shown in Table 7, AROID achieves a marginal improvement over this baseline in terms of accuracy and robustness, indicating that AROID remains effective even in data-rich settings. However, it is observed that the performance improvement provided by AROID diminishes when compared to results without the additional 50 M data. This reduction occurs because the robust overfitting in the baseline is largely mitigated by the additional data, and since AROID enhances adversarial training by alleviating robust overfitting, the scope for further improvement by AROID is consequently reduced.

Although the benefit of data augmentation diminishes when a large amount of synthetic data is incorporated for training on CIFAR10, this approach may not be as effective on more complex datasets such as ImageNet. As observed in Azizi et al. (2023), increasing synthetic ImageNet data beyond a certain limit (around 1.2M synthetic images) degrades model performance in high-resolution settings (256×256 and 1024×1024 pixels), while it consistently provides benefits in low-resolution setting (64×64 pixels). This degradation at high resolutions may be due to greater bias in the model and/or lower quality in the generated images at higher resolutions.

4.8 Generalization to ImageNet

To further test the generalizability and scalability of our method to a large-scale dataset, we train AROID on ImageNet (Deng et al., 2009) with ConvNeXt-T (Liu et al., 2022). Some DA methods are missing in this comparison due to limited computational resources (explained in Section D.2). As shown in Table 8, AROID significantly improves robustness over the baseline by 4.18% and AutoAugment by 2.6%. It also achieves the highest accuracy among the tested methods. Overall, AROID is able to scale and generalize to ImageNet.

Table 8 The result of AROID on ImageNet with ConvNeXt-T

DA method	Accuracy	Robustness
RandomCrop	<u>71.22</u>	36.22
AutoAugment	70.42	<u>37.80</u>
AROID (ours)	**71.62**	**40.40**

Bold value indicates the best performance, and Underline value indicates the second best performance

The AROID hyperparameters were set to $\lambda = 0.7$, $\beta = 2$, $(l, u) = (0.8, 4.0)$, $T = 20$ and $K = 4$. As we did not have sufficient computational resources to fully optimize these hyperparameters on ImageNet performance is likely to be suboptimal and falls-short of the state-of-the-art result (Singh et al., 2023). It has been observed in Singh et al. (2023) that adversarial training on ImageNet prefers heavy data augmentation that is composed of RandAugment (Cubuk et al., 2020), CutMix, MixUp and Random Erasing. DA operations like CutMix and MixUp are not included in our DA search space. Incorporating these operations into our search space is thus expected to boost the performance of our method on ImageNet. We leave the exploration of this enhancement to the future.

4.9 Performance on Common Corruption Datasets

This section assesses the generalization capability of the proposed method under input data distribution shifts, known as Out-Of-Distribution (OOD) testing. Following Kireev et al. (2022), we trained models on the CIFAR10 training set and evaluated them on CIFAR10-C (Hendrycks & Dietterich,

Table 9 The performance of various DA methods on the common corruption dataset CIFAR10-C for WRN34-10

Method	CIFAR10		CIFAR10-C	
	Acc.	Rob.	Acc.	Rob.
RandomCrop	85.83	52.26	76.70	36.69
Cutout	86.95	52.89	76.46	35.97
CutMix	86.88	53.38	77.48	36.91
AutoAugment	87.71	54.60	78.30	36.68
TrivialAugment	87.35	53.86	78.42	37.99
IDBH	88.61	55.29	79.37	38.15
AROID (ours)	**88.99**	**55.91**	**80.61**	**39.72**

Bold value indicates the best performance, and Underline value indicates the second best performance
Models were trained on CIFAR10 training set

2019). CIFAR10-C is created by applying 15 types of common visual corruptions to the CIFAR10 test set, representing visual corruption shifts encountered in the wild.

In Kireev et al. (2022), only clean accuracy was evaluated on CIFAR10-C, focusing on the efficacy of adversarial training in improving robustness against common corruptions. However, this study emphasizes adversarial robustness. A recent study suggested that adversarial robustness is highly vulnerable to input distribution shifts (Li et al., 2024). Therefore, we also evaluated adversarial robustness on CIFAR10-C by conducting AutoAttack on the CIFAR10-C data.

As shown in Table 9, our proposed method achieves the highest accuracy and robustness among all competitive data augmentation methods, indicating excellent OOD generalization ability for both clean and robust performance under common corruption distribution shifts.

4.10 Robustness Evaluation with More Attacks

To further ensure our robustness evaluation is reliable, we additionally evaluate AROID and other related works using three more adversarial attacks PGD (Madry et al., 2018), CW (Carlini & Wagner, 2017) and JITTER (Schwinn et al., 2023). From the results shown in Table 10 it can be seen that AROID is consistently superior under various adversarial attacks.

4.11 Data Scaling Versus Model Scaling

This section compares the effectiveness of scaling up data (our method) versus scaling up the model in enhancing adversarial training. To test this, we trained AROID using the WRN34-10 model architecture (depth of 34 and widening factor of 10) and compared it to WRN34-12 and WRN46-10 architectures trained with RandomCrop DA. WRN34-12 and WRN46-10 were chosen because they have approximately 44% and 42% more parameters, respectively, than WRN34-10, which is comparable to the worst-case extra computational overhead, 43.6%, caused by AROID.

As shown in Table 11, AROID with WRN34-10 achieved the highest accuracy and robustness, greatly outperforming RandomCrop even when larger models were used. This suggests that **optimizing data augmentation, when implemented correctly, can be more effective than merely scaling up the model to boost performance**. The issue with RandomCrop and larger models is that, as indicated by the large gap between best and end robustness, scaling up models cannot effectively mitigate robust overfitting, resulting in poor generalization of robustness.

4.12 Enlarging Policy Search Space

This section assesses if enlarging policy search space can enhance AROID. We conducted tests by adding CutMix to our policy search space as an additional transformation to be sampled and applied after the dropout transformation (please refer to Sect. 3.1 for the specification of data augmentation policy structure). CutMix was chosen due to its effectiveness in adversarial training when combined with SWA (Rebuffi et al., 2021).

As shown in Table 12, **the inclusion of CutMix, compared to the original data augmentation space, results in**

Table 10 Robustness evaluation against more adversarial attacks

DA methods	CIFAR10+WRN34-10					Imagenette+ViT-B/16				
	Clean	AA	PGD	CW	JITTER	Clean	AA	PGD	CW	JITTER
RandomCrop	85.8	52.2	55.5	54.2	53.5	92.7	66.4	68.1	68.4	68.8
Cutout	86.9	52.8	55.3	55.0	54.6	93.2	67.2	68.4	68.6	69.4
CutMix	86.8	53.3	**60.1**	56.9	56.4	93.8	70.2	**73.1**	71.8	72.2
AutoAugment	87.7	54.6	58.8	56.3	55.6	95.1	67.6	68.9	69.8	70.6
TrivialAugment	87.3	53.8	57.4	55.2	55.4	**95.2**	69.0	70.9	70.6	71.5
IDBH	88.6	55.2	58.2	57.3	56.9	95.2	69.9	70.2	70.8	71.6
AROID (ours)	**88.9**	**55.9**	59.6	**58.1**	**57.6**	94.8	**71.3**	71.8	**72.8**	**73.1**

Bold value indicates the best performance, and Underline value indicates the second best performance
PGD uses 50 steps and 10 restarts. CW and JITTER use 100 steps. Note that the abnormally superior PGD robustness but worse against other attacks of CutMix suggest a false security caused by obfuscated gradients

Table 11 The performance of baseline RandomCrop with larger models on CIFAR10

Data augmentation	Model	Model size (M)	Accuracy (%)			Robustness (%)		
			Best	End	Diff.	Best	End	Diff.
AROID	WRN34-10	46.2	**88.99**	**89.29**	**-0.30**	**55.91**	**55.00**	**0.91**
RandomCrop	WRN34-10	46.2	85.83	86.21	-0.38	52.26	46.63	5.63
RandomCrop	WRN34-12	66.5	86.65	86.45	0.20	52.46	48.34	4.12
RandomCrop	WRN46-10	65.5	86.61	86.38	0.23	52.98	47.63	5.35

Bold value indicates the best performance, and Underline value indicates the second best performance

Table 12 The performance of AROID with the original and the enlarged (with CutMix added) data augmentation space with and without SWA for WRN34-10 on CIFAR10

AT method	DA space	Accuracy (%)			Robustness (%)		
		Best	End	Diff.	Best	End	Diff.
AT	Original	89.50	89.59	**−0.09**	55.56	53.33	2.23
	Original+CutMix	88.93	89.46	−0.53	56.44	55.83	0.62
AT-SWA	Original	88.71	**90.40**	−1.69	57.31	55.05	2.26
	Original+CutMix	**89.52**	90.02	−0.50	**57.32**	**57.14**	**0.18**

Bold value indicates the best performance, and Underline value indicates the second best performance
Models were trained for 400 epochs, in contrast to 200 epochs used in Table 3, to better demonstrate the effect of adding CutMix in reducing robust overfitting

reduced robust overfitting and improved best and end robustness, regardless of whether it is combined with SWA or not. Additionally, incorporating CutMix even leads to a boost in best accuracy when combined with SWA. One possible account for this improvement is that the addition of CutMix increases the diversity of data augmentation in the learned policy, thereby mitigating robust overfitting and enhancing robust generalization (the reasons why diverse data augmentation mitigates robust overfitting are explained in Sect. 3.2.1).

However, it is important to note that not all data augmentation methods yield such benefits. The impact of incorporating additional data augmentation methods into the policy search space is specific to the nature of the augmentation techniques themselves. Toxic data augmentation methods, as observed in Cubuk et al. (2020), may not enhance, and in some cases, may even impair the performance of AROID if added to the search space. Overall, AROID can indeed benefit from an enlarged search space if implemented appropriately.

4.13 Ablation Study

This section verifies the sensitivity of our method to its hyper-parameters and several design choices. The experiments were conducted on CIFAR10 with PRN18 and Imagenette with ViT-B/16. The default values of hyperparameters are the ones marked in green in Fig. 3.

4.13.1 Hyperparameters

Policy update frequency K. Figures 3j and l show that the highest accuracy and robustness were achieved when $K = 5$, i.e., the lowest frequency under the test. This implies that AT benefits from a more "up-to-date" DA. Furthermore, it seems possible to trade accuracy for efficiency by choosing a larger value of K (up to 20) while maintaining similarly high robustness. In general, the accuracy and robustness of our method declines with lower policy update frequency.

Number of trajectories T. Figure 3i and k show that high accuracy and robustness are achieved around $T = 8$. This suggests that (1) there is a minimum requirement on the amount of trajectories for our policy gradient estimator to be accurate and, (2) our method may not benefit from increasing T beyond 8.

Strength of Affinity λ. As shown in Fig. 3a and c, robustness first increases and then decreases within the tested range of value. This is consistent with the prior that AT benefits from appropriate hardness but degrade if data augmentations are overly hard (Li & Spratling, 2023c).

Strength of Diversity β. The performance within the tested range of value is close in Fig. 3b and d, suggesting that the performance of AROID is not sensitive to the value of β. Nevertheless, this does not imply that Diversity is unnecessary in our policy learning. On the contrary, it plays an important role in policy learning as shown in Sect. 4.13.2.

Summary. We observe that, within the tested value range, hyper-parameters like λ, β, T and K have a quite similar trend in both settings, while the lower limit l (Fig. 3e, g) and upper limit u (Fig. 3f, h) in the diversity objective shows slightly different trends between the two settings. Despite the slightly different behaviors of a few hyper-parameters, the optimal value of hyper-parameters is observed to transfer across these two settings, i.e., they achieve reasonably good performance with a similar set of hyper-parameter values $T = 8$, $K = 5$, $l = 0.8/0.9$, $u = 4$, $\lambda = 0.3$, $\beta = 0.8$. We also find this setting transfers well across different AT methods of

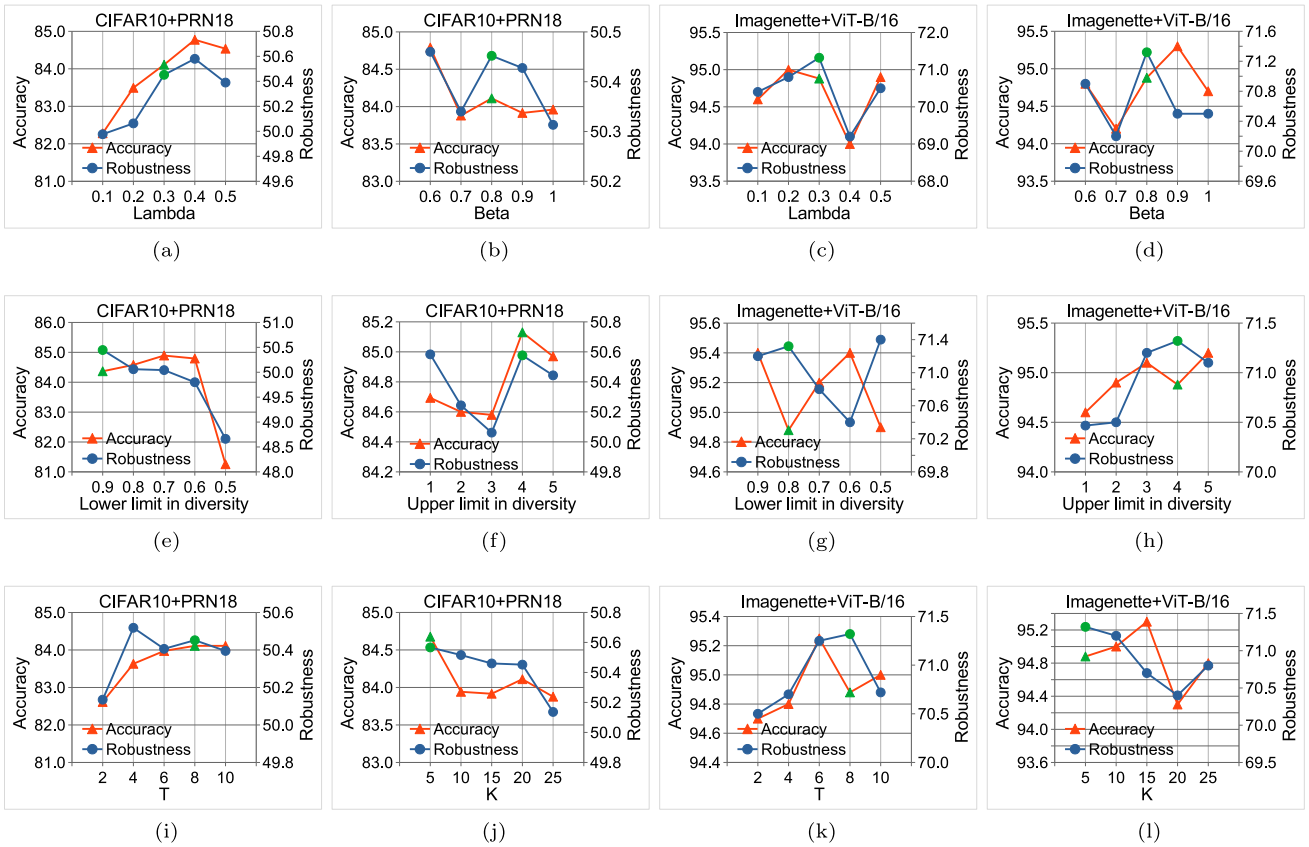

Fig. 3 Ablation study of hyper-parameters λ, β, l, u, T and K for CIFAR10 with PRN18 (even rows) and Imagenette with ViT-B/16 (odd rows). The selected value for each hyper-parameter is marked green color

PGD, SCORE and TRADES since we can only tune the value of λ while keep the rest unchanged to achieve reasonably good performance and outperform the other compared data augmentations.

4.13.2 Policy Learning Objectives

This section conducts an ablation study to evaluate the effect of each proposed policy learning objective on the performance of AROID. As shown in Table 13, **removing any single policy learning objective leads to a considerable drop in both accuracy and robustness, indicating that each objective is crucial for learning an effective data augmentation policy**. Particularly, we observed that when Diversity is removed by setting $\beta = 0$, accuracy drops from 84.68 to 73.88%, and robustness drops from 50.57 to 22.24%. Without Diversity constraint, the policy network's training failed because the output policy distribution became concentrated on a few sub-policies, assigning zero probabilities to the remaining ones. The REINFORCE method could not recover from this situation because it no longer explored other options. This underscores the importance of maintaining a certain level of Diversity constraint in our policy learning.

Table 13 The impact of removing each policy learning objective on the performance of AROID for PRN18 on CIFAR10

Policy objectives	Accuracy (%)	Robustness (%)
AROID	**84.68**	**50.57**
- Vulnerability	83.50 (−1.18)	49.95 (−0.62)
- Affinity	82.03 (−2.65)	49.41 (−1.16)
- Diversity	73.88 (−10.8)	22.47 (−28.1)

Bold value indicates the best performance
The performance drop is given in the bracket

However, no clear benefit is observed as this constraint is further strengthened by raising β, as shown in Fig. 3b and d.

4.13.3 Policy Model Architecture

Interestingly, we observed in Table 14 that for CIFAR10 a relatively small model WideResNet10-1 (a WideResNets with depth 10 and widening factor 1) with 0.08M parameters is sufficient for learning the DA policy for a relatively large target model PRN18 with 11.17M parameters and further increasing capacity beyond this scale, even 100x, does not benefit either accuracy or robustness. Therefore, the policy model can be much smaller than the target model.

Table 14 Comparison of the various policy model backbone architectures on CIFAR10 with a target model of PRN18

Model	Size (M)	Clean	AA
WRN10-1	0.08	84.16	50.25
WRN22-1	0.27	84.32	**50.57**
WRN34-1	0.47	**84.73**	50.38
WRN70-1	1.05	84.04	50.28
PRN18	**11.17**	84.68	**50.57**

Bold value indicates the best performance, and Underline value indicates the second best performance

Table 15 Comparison of uniform sampling from AROID DA space on CIFAR10 with PRN18

DA	Clean	AA
RandomCrop	82.50	48.21
Uniform	81.00	49.18
AROID	**84.68**	**50.57**

Bold value indicates the best performance, and Underline value indicates the second best performance

4.13.4 Uniform Sampling

We performed AT using data augmentations uniformly sampled from AROID's data augmentation space. The results are labeled Uniform in Table 15. As shown in the table, AROID significantly improves accuracy and robustness over its uniformly sampled counterpart suggesting the necessity of optimizing the data augmentation policy.

4.14 Analysis of Learned DA Policies

This section first analyzes the dynamics of the proposed policy learning objectives during training (Sect. 4.14.1). It then visualizes the learned data augmentation policies sampled over a course of training (Sect. 4.14.2). Last, it visualizes some image samples transformed by the learned data augmentation policies (Sect. 4.14.3).

4.14.1 Progression of Policy Learning Objectives

To understand the dynamics of the learned data augmentation policy, Fig. 4 visualizes the progression of the three proposed policy learning objectives throughout the AROID training process. Generally, Vulnerability represents the adversarial vulnerability of the augmented data, Affinity reflects the distribution shift caused by data augmentation, and Diversity is negatively correlated with the diversity of data augmentation (lower Diversity implies greater diversity). It is observed that during training, Vulnerability and Affinity increase while Diversity decreases. These trends suggest that the data augmentation sampled from the learned policies becomes progressively harder, in terms of both adversarial vulnerability and distribution shift, and more diverse throughout the training process. This aligns with the goal of our policy learning as described in Eq. (8) to encourage an increase in Vulnerability while regularizing Affinity and Diversity to decrease. It is important to note that an increase, rather than a decrease, is observed in the Affinity loss because Affinity was regularized with a decaying strength (in this case 0.4, 0.2, 0.1).

4.14.2 Visualization of Learned DA Policies

Figure 5 visualizes the learned distribution of DAs for different, randomly sampled, data instances. Instance-wise variation of the learned DA policy is visible for the Color/Shape augmentations (Fig. 5c) and evident for the Dropout augmentations (Fig. 5d), but subtle in the rest (Fig. 5a, b). Note that even for the different data instances from the same class (e.g., instances 4, 7, 10 from the class "frog"), the learned DA distributions can still differ considerably (Fig. 5d). This confirms that (1) AROID is able to capture and meet the varied demand of augmentations from different data instances, and (2) such demand exists for some, but not all, augmentations. These observations may explain why many instance-agnostic DA

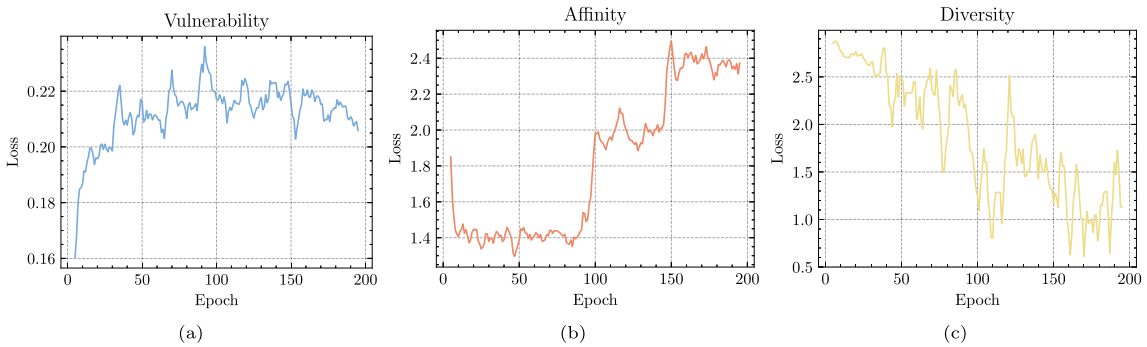

Fig. 4 The progression of the three proposed policy learning objectives throughout the AROID training process on CIFAR10 for WRN34-10. Lines are smoothed with a moving average over 5 epochs for improved clarity

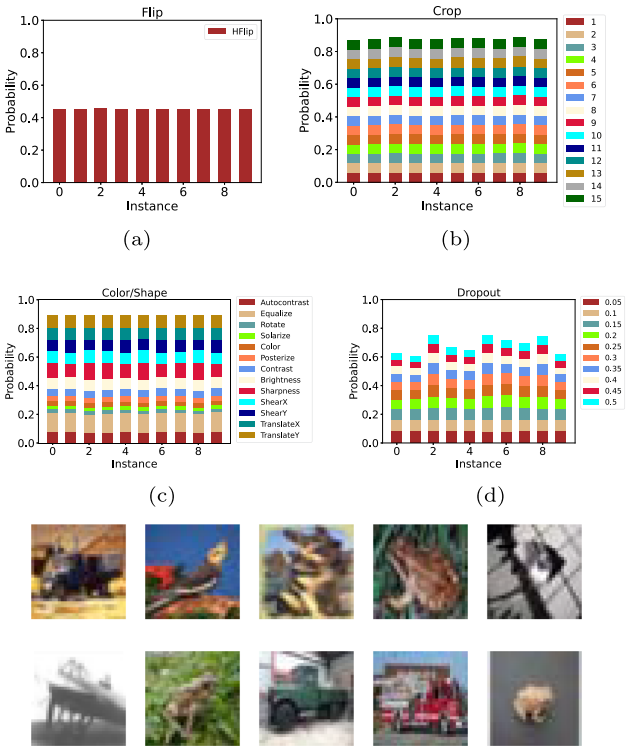

Fig. 5 Visualization of the learned DA policies, applied to ten images randomly sampled from CIFAR10 training set, for the Flip, Crop, Color/Shape and Dropout types of augmentations. The policy model is resumed from a checkpoint saved at the end of 110th epoch when training a WRN34-10 model on CIFAR10 (following the training setting as specified in Section D). The sampled ten images are visualized at the bottom in the order of the x-axis in the above bar-charts. The chance of applying no transformation (Identity) is the gap between the colored bar and the top (i.e., score of 1.0). In the Color/Shape group, the probabilities of different magnitudes are not shown separately, but are summed to get the overall probability of a transformation

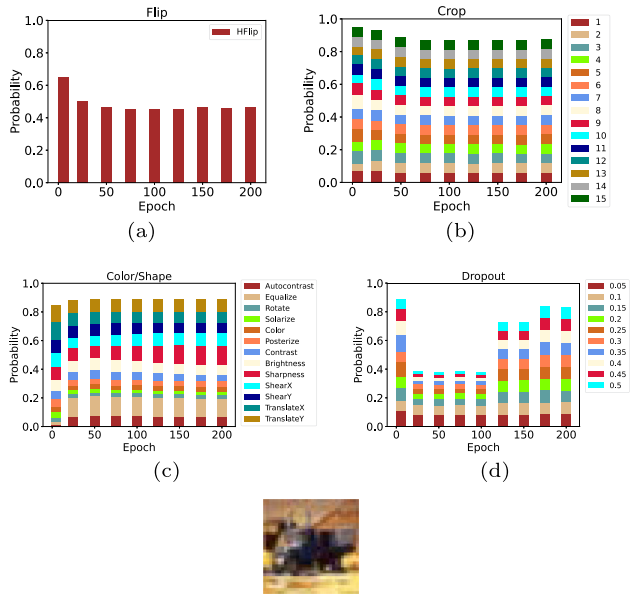

Fig. 6 Visualization of how the learned DA policies evolve as training progresses. The same, randomly sampled, image (visualized at the bottom) was used across epochs (5, 25, 50, 75, 100, 125, 150, 175, 200) to produce the policies. The first bar in each sub-figure corresponds to the epoch 5 and describes the initial state of the policy model (training of policy model starts from epoch 5). For each bar in the figures, the policy model was resumed from the checkpoint saved at the corresponding epoch (x-axis) in the same course of training. The chance of applying no transformation (Identity) is the gap between the colored bar and the top (i.e., the score of 1.0). In the Color/Shape group, the probabilities of different magnitudes are not shown separately, but are summed to get the overall probability of a transformation

methods such as IDBH, despite being inferior to ours, still work reasonably well (see Table 1).

It was also observed in Fig. 6 that the learned DA policy for the same data instance evolved as training progressed. In the Color/Shape group (Fig. 6c), augmentations like Sharpness became observably more likely to be selected while others such as ShearY became less probable as training continued. Dropout (i.e. Erasing; Fig. 6d) particularly with large magnitudes was rarely applied prior to 100th epoch, i.e., the first decay of learning rate. The possibility of applying Crop (i.e. Cropshift; Fig. 6b) and Flip (i.e. HorizontalFlip; Fig. 6a) first dropped until the first decay of learning rate and then stayed nearly constant afterwards.

Consistent to the previous findings on ST (Cubuk et al., 2019) and harmful augmentations (Rebuffi et al., 2021), we observed that AT on CIFAR10 favored mostly color-based augmentations like Equalize and Sharpness and disfavored geometric augmentations like Rotate and harmful augmentations like Solarize and Posterize (see both Figs. 5c, 6c). This verifies the effectiveness of our DA policy learning algorithm.

4.14.3 Visualization of Augmented Data Samples

Figure 7 depicts 20 pairs of original and augmented data samples from CIFAR10. The visualization demonstrates that **our method effectively enhances the diversity of augmented data samples**. While the original and augmented data samples are paired here in a one-to-one manner, the learned policy enables the generation of a much larger variety of distinct augmented data.

5 Conclusions

This work introduces an approach, dubbed AROID, to efficiently learn online, instance-wise, DA policies for improved robust generalization in AT. AROID is the first automated DA method specifically for AT. Extensive experiments show its superiority over both alternative DA methods and contemporary AT methods in terms of accuracy and robustness. AROID has also significantly reduces the cost of policy

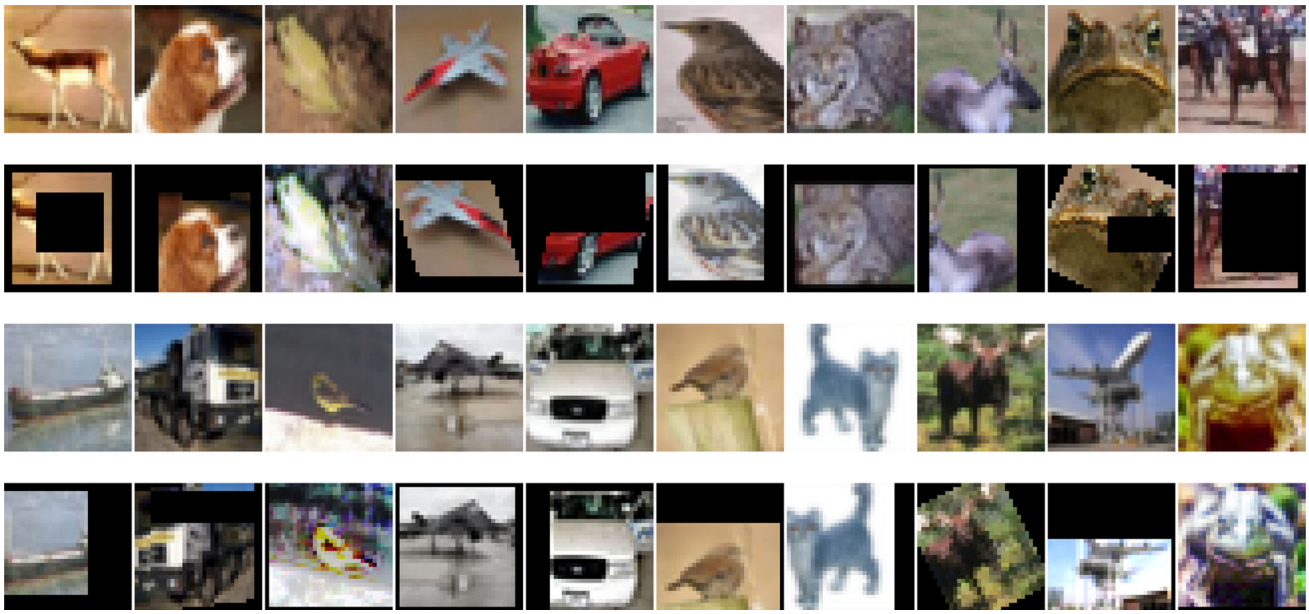

Fig. 7 Visualization of 20 randomly-sampled pairs of original (odd rows) and augmented (even rows) samples from CIFAR10. The policy model is the same as that used for Fig. 5

search making automated data augmentation practical to use for adversarial training, even for large datasets. AROID can be also used in an offline mode to further save on computation. The learned DA policies are visualized to verify the effectiveness of AROID and understand the preference of AT for DA.

However, AROID has some limitations as well. First, despite being more efficient than IDBH, it still adds extra computational burden to training, unless AROID-T is used. This could harm its scalability to larger datasets and model architectures. Second, the Diversity objective enforces a minimal chance (set by the lower limit) of applying harmful transformations and/or harmful magnitudes if they are included in the search space. This constrains the ability of AROID to explore a wider (less filtered) search space. Future works could investigate more efficient AutoML algorithms for learning DA policies for AT, and design new policy learning objectives to reduce the number of hyperparameters and alleviate the side-effect of Diversity.

Appendix A DA Search Space

Table 16 shows the complete DA search space used by AROID. For Color/Shape group, we adopted the same operations as RandAugment's, but discretize the range of magnitudes for each operation into 10 even values if possible. For Erasing in Dropout group, the magnitude corresponds to the scale (the proportion of erased area against input image), while the aspect ratio (of erased area) is uniformly sampled from range $(0.3, 3.3)$. The search space only defines the operations and their magnitudes, while the probabilities of applying these operations are learned by AROID.

Appendix B Derivation

This section discusses how we derive the gradients of Hardness metric w.r.t. the parameters of the policy model:

$$\frac{\partial \mathbb{E}_{i \in B} \mathcal{L}_{hrd}(\boldsymbol{x}_i)}{\partial \boldsymbol{\theta}_{plc}} \tag{B1}$$

First, we rewrite Eq. (B1) as below, so that we can focus on the gradient derivation part.

$$\frac{1}{B} \sum_{i=1}^{B} \frac{\partial \mathcal{L}_{hrd}(\boldsymbol{x}_i)}{\partial \boldsymbol{\theta}_{plc}} \tag{B2}$$

Next, to apply the REINFORCE algorithm, we substitute the gradient of the $\mathcal{L}_{hrd}$ for a sampled trajectory in Eq. (B2) with the gradient of the expected $\mathcal{L}_{hrd}$ for multiple sampled trajectories as

$$\frac{1}{B} \sum_{i=1}^{B} \frac{\partial \mathbb{E}_{t \in T} \mathcal{L}_{hrd}^{(t)}(\boldsymbol{x}_i)}{\partial \boldsymbol{\theta}_{plc}} \tag{B3}$$

Table 16 Augmentation space

Flip			Crop			Color/shape			Dropout		
Operations	Magnitudes	Count	Operations	Magnitudes	Count	Operations	Magnitudes	Count	Operations	Magnitudes	Count
Identity	–	1	Identity	–	1	Identity	–	1	Identity	–	1
Horiz. Flip	–	1	Cropshift	1, 2, 3, 4, 5	15	Autocontrast	–	1	Erasing	.05, .10	10
				6, 7, 8, 9		Equalize	–	1		.15, .20	
				10, 11, 12		Posterize	4, 5, 6, 7, 8	5		.25, .30	
				13, 14, 15		Solarize	25, 51, 76, 102	10		.35, .40	
							128, 153, 179			.45, .50	
							204, 230, 256				
						Rotate	3, 6, 9, 12, 15	10			
							18, 21, 24, 27, 30				
						ShearX	.03, .06, .09	10			
							.12, .15, .18				
							.21, .24, .27, .30				
						ShearY	.03, .06, .09	10			
							.12, .15, .18				
							.21, .24, .27, .30				
						TranslateX	1, 2, 3, 4, 5	10			
							6, 7, 8, 9, 10				
						TranslateY	1, 2, 3, 4, 5	10			
							6, 7, 8, 9, 10				
						Color	.28, .46, .64, .82	10			
							1.0, 1.18, 1.36				
							1.54, 1.72, 1.9				
						Contrast	.28, .46, .64, .82	10			
							1.0, 1.18, 1.36				
							1.54, 1.72, 1.9				
						Brightness	.28, .46, .64, .82	10			
							1.0, 1.18, 1.36				
							1.54, 1.72, 1.9				
						Sharpness	.28, .46, .64, .82	10			
							1.0, 1.18, 1.36				
							1.54, 1.72, 1.9				

By applying the REINFORCE algorithm, we have (batch averaging is omitted for simplicity)

$$\frac{\partial \mathbb{E}_{t\in T}\mathcal{L}^{(t)}_{hrd}(\boldsymbol{x}_i)}{\partial \boldsymbol{\theta}_{plc}}$$

$$= \frac{\partial \sum_{t=1}^{T}\mathcal{P}_{(t)}(\boldsymbol{x}_i)\mathcal{L}^{(t)}_{hrd}(\boldsymbol{x}_i)}{\partial \boldsymbol{\theta}_{plc}} \tag{B4}$$

$$= \sum_{t=1}^{T}\frac{\partial \mathcal{P}_{(t)}(\boldsymbol{x}_i)}{\partial \boldsymbol{\theta}_{plc}}\mathcal{L}^{(t)}_{hrd}(\boldsymbol{x}_i) \tag{B5}$$

$$= \sum_{t=1}^{T}\mathcal{P}_{(t)}(\boldsymbol{x}_i)\frac{\partial \log(\mathcal{P}_{(t)}(\boldsymbol{x}_i))}{\partial \boldsymbol{\theta}_{plc}}\mathcal{L}^{(t)}_{hrd}(\boldsymbol{x}_i) \tag{B6}$$

$$= \mathbb{E}_{i\in T}\frac{\partial \log(\mathcal{P}_{(t)}(\boldsymbol{x}_i))}{\partial \boldsymbol{\theta}_{plc}}\mathcal{L}^{(t)}_{hrd}(\boldsymbol{x}_i) \tag{B7}$$

$\mathcal{P}_{(t)}(\boldsymbol{x}_i)$ is the probability of sampled trajectory. Following the previous practices (Zhang et al., 2020; Lin et al., 2019; Jia et al., 2022), we approximate Eq. (B7) as

$$\frac{1}{T}\sum_{t=1}^{T}\frac{\partial \log(\mathcal{P}_{(t)}(\boldsymbol{x}_i))}{\partial \boldsymbol{\theta}_{plc}}\mathcal{L}^{(t)}_{hrd}(\boldsymbol{x}_i) \tag{B8}$$

Next, by expanding $\mathcal{P}_{(t)} = \prod_{h=1}^{H} p^h_{(t)}$, we have

$$\frac{1}{T}\sum_{t=1}^{T}\frac{\partial \log(\prod_{h=1}^{H} p^h_{(t)}(\boldsymbol{x}_i;\boldsymbol{\theta}_{plc}))}{\partial \boldsymbol{\theta}_{plc}}\mathcal{L}^{(t)}_{hrd}(\boldsymbol{x}_i) \tag{B9}$$

$$\approx \frac{1}{T}\sum_{t=1}^{T}\frac{\partial \sum_{h=1}^{H}\log(p^h_{(t)}(\boldsymbol{x}_i;\boldsymbol{\theta}_{plc}))}{\partial \boldsymbol{\theta}_{plc}}\mathcal{L}^{(t)}_{hrd}(\boldsymbol{x}_i) \tag{B10}$$

$$\approx \frac{1}{T}\sum_{t=1}^{T}\sum_{h=1}^{H}\frac{\partial \log(p^h_{(t)}(\boldsymbol{x}_i;\boldsymbol{\theta}_{plc}))}{\partial \boldsymbol{\theta}_{plc}}\mathcal{L}^{(t)}_{hrd}(\boldsymbol{x}_i) \tag{B11}$$

To reduce the variance of gradient estimation, we apply the baseline trick by subtracting mean value, $\tilde{\mathcal{L}_{hrd}} = \frac{1}{T}\sum_{t=1}^{T}\mathcal{L}^{(t)}_{hrd}(\boldsymbol{x}_i)$, from $\mathcal{L}^{(t)}_{hrd}$ as

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{h=1}^{H}\frac{\partial \log(p^h_{(t)}(\boldsymbol{x}_i;\boldsymbol{\theta}_{plc}))}{\partial \boldsymbol{\theta}_{plc}}[\mathcal{L}^{(t)}_{hrd}(\boldsymbol{x}_i) - \tilde{\mathcal{L}_{hrd}}] \tag{B12}$$

Eventually, by adding back the batch averaging, we have our ultimate form of gradients as

$$\frac{1}{B\cdot T}\sum_{i=1}^{B}\sum_{t=1}^{T}\sum_{h=1}^{H}\frac{\partial \log(p^h_{(t)}(\boldsymbol{x}_i))}{\partial \boldsymbol{\theta}_{plc}}[\mathcal{L}^{(t)}_{hrd}(\boldsymbol{x}_i) - \tilde{\mathcal{L}_{hrd}}] \tag{B13}$$

Appendix C Efficiency Analysis

The efficiency of AROID is analyzed here. $F_t/F_p/F_a$ and $B_t/B_p/B_a$ denote the cost of forward and backward pass on target/policy/affinity model respectively. For each iteration of updating policy model, the major overhead is

- Predict DA distribution: 1 F_p
- Vulnerability: for each of T trajectories, 2 $(F_t + B_t)$ to generate adversarial examples and 1 F_t to calculate loss. Overall, $(3F_t + 2B_t)T$
- Affinity: 1 F_a to calculate the loss of original data which is shared by all T trajectories. 1 F_a to calculate the loss of augmented data for each of T trajectories. Overall, $(F_aT + F_a)$
- Diversity: the calculation of diversity loss adds negligible overhead and does not require F or B
- Update policy model: 1 B_p

To sum up, one iteration of policy update costs

$$(3F_t + 2B_t)T + (F_aT + F_a) + F_p + B_p \tag{C14}$$

Policy model is updated every K iterations of target model, so the averaged policy learning cost per iteration of target model training is

$$[(3F_t + 2B_t)T + (F_aT + F_a) + F_p + B_p]/K \tag{C15}$$

The overall overhead of AROID is learning cost plus 1 F_p for every iteration of target model to sample DA, so

$$[(3F_t + 2B_t)T + (F_aT + F_a) + F_p + B_p]/K + F_p \tag{C16}$$

In worst case, policy and affinity models use the same architecture as target model, so the cost is

$$[(4T + 2)/K + 1]F_t + (2T + 1)B_t/K \tag{C17}$$

The most expensive setting we use is $T = 8$ and $K = 5$, so it costs $7.8F_t + 3.4B_t$ roughly, assuming $2F_t = 1B_t$, $4.8(F_t + B_t)$ in addition to $11(F_t + B_t)$ of underlying PGD10 AT. Overall, in worst case, AROID adds about 43.6% extra computation to baseline AT. For a cheaper setting $T = 4$ and $K = 20$, the overhead is roughly $1.9F_t + 0.45B_t$ about 10% more than baseline AT.

Appendix D Experimental Set-ups

D.1 Configuration of AROID

Vulnerability objective was calculated based on PGD2 with a step size of 2/255 except that PGD1 with a step size of 4/255 for ImageNet. The affinity models used the same architecture as the target model. The affinity models were pre-trained using ST with the same settings as their AT trained counterparts yet with no augmentation. Early stopping was used if training accuracy was close to 100%. The policy model was trained using SGD with a constant learning rate (0.001 by default while 0.1 for Imagenette due to the reduced number of training epochs) and the same momentum as the target optimizer's. Gradient clipping was applied to stabilize the training of the policy model. In the initial five epochs of training, we did not train the policy model nor apply it to augment the data (no augmentation at all was applied) since the target model changed rapidly.

D.2 Configuration of Compared DA Methods

AutoAugment was parameterized as in Cubuk et al. (2019) since we did not have sufficient resource to optimize. For AutoAugment, augmentations were applied in the order of HorizontalFlip-RandomCrop-AutoAugment-Cutout (16×16) as in Cubuk et al. (2019). TrivialAugment is parameter-free so no tuning was needed. For TrivialAugment, augmentations were applied in the order of HorizontalFlip-RandomCrop-TrivialAugment-Cutout (16×16)) as in Müller and Hutter (2021). For CutMix, $\alpha = 0.25$ and $\beta = 1$ on CIFAR10 as optimized in Li and Spratling (2023c); $\alpha = 1$ and $\beta = 1$ on Imagenette as suggested in Yun et al. (2019). For Cutout, the size of cut-out area was 20×20 on all three datasets as in Li and Spratling (2023c). Cutout and CutMix were applied with the default (baseline) augmentations in the order of HorizontalFlip-RandomCrop-Cutout and -CutMix respectively on CIFAR10 and Imagenette. For IDBH, IDBH[strong]-CIFAR10 was used.

We only compare our method against the baseline and AutoAugment on ImageNet. AutoAugment is selected because it is one of the two methods closest to AROID and has a pre-optimized version for ImageNet while the other closest work IDBH doesn't. Due to the tremendous cost of conducting AT on ImageNet and the limit of our computational resource, we can't optimize other DA methods for AT on ImageNet so they are not included to avoid unfair comparison. In fact, like most other researchers, we don't have enough time and resource to train all competitive DA methods even without re-optimization of hyperparameters.

D.3 Configuration of Compared State-of-the-art Robust Training Methods

We only re-implemented the algorithms of SWA and AWP to report the result based on our runs, while the result of the others including MART, MART-AWP, SEAT, LAT-AT and LAS-AWP were copied directly from their original works except that the result of MART was copied from (Wu et al., 2020) for a better aligned training setting. SWA was implemented as in Rebuffi et al. (2021) with a decay rate of $\tau = 0.999$. AWP was configured as in (Wu et al., 2020) with $\beta = 0.005$. Note that the same configurations of SWA and AWP were used to train with baseline DA and AROID.

Acknowledgements The authors gratefully acknowledge use of the King's Computational Research, Engineering and Technology Environment (CREATE) for carrying out the experiments described in this paper.

Author Contributions Conceptualization: LL; Methodology: LL and MS; Software: LL; Validation: LL, JQ and MS; Formal analysis and investigation: LL; Visualization: LL and JQ; Writing—original draft preparation: LL and JQ; Writing—review and editing: MS and JQ; Funding acquisition: LL; Supervision: MS.

Funding This work was funded by a scholarship from the King's - China Scholarship Council (K-CSC).

Availability of data and material All datasets used are publicly available.

Code availability Code is available in the supplementary material and will be published on Github once the paper is accepted for publication.

Declarations

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

Ethics approval Not applicable.

Consent to participate Not applicable.

Consent for publication Not applicable.

References

Addepalli, S., Jain, S., & Radhakrishnan, V. B. (2022). Efficient and effective augmentation strategy for adversarial training. In *Neural information processing systems (NeurIPS)*.

Athalye, A., Carlini, N., & Wagner, D. (2018). Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International conference on machine learning (ICML)*.

Azizi, S., Kornblith, S., Saharia, C., Norouzi, M., & Fleet, D. J. (2023). Synthetic data from diffusion models improves imagenet classification. In *Transactions on machine learning research (TMLR)*.

Carlini, N., & Wagner, D. (2017). Towards evaluating the robustness of neural networks. In *IEEE symposium on security and privacy (SP)*.

Carmon, Y., Raghunathan, A., Schmidt, L., Duchi, J. C., & Liang, P. S. (2019). Unlabeled data improves adversarial robustness. In *Neural information processing systems (NeurIPS)*.

Chen, T., Zhang, Z., Liu, S., Chang, S., & Wang, Z. (2021). Robust Overfitting may be mitigated by properly learned smoothening. In *International conference on learning representations (ICLR)*.

Cheung, T.-H., & Yeung, D.-Y. (2022). AdaAug: Learning class- and instance-adaptive data augmentation policies. In *International conference on learning representations (ICLR)*.

Croce, F., Andriushchenko, M., Sehwag, V., Debenedetti, E., Flammarion, N., Chiang, M., & Hein, M. (2021). RobustBench: A standardized adversarial robustness benchmark. In *Neural information processing systems (NeurIPS)*.

Croce, F., & Hein, M. (2020). Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International conference on machine learning (ICML)*.

Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., & Le, Q. V. (2019). AutoAugment: Learning augmentation strategies from data. In *IEEE/CVF conference on computer vision and pattern recognition (CVPR)*.

Cubuk, E. D., Zoph, B., Shlens, J., & Le, Q. V. (2020). Randaugment: Practical automated data augmentation with a reduced search space. In *Neural information processing systems (NeurIPS)*.

Deng, J., Dong, W., Socher, R., Li, L., Kai, L., & Li, F.-F. (2009). ImageNet: A large-scale hierarchical image database. In *IEEE/CVF conference on computer vision and pattern recognition (CVPR)*.

DeVries, T., & Taylor, G. W. (2017). Improved regularization of convolutional neural networks with cutout. arXiv.

Dong, Y., Xu, K., Yang, X., Pang, T., Deng, Z., Su, H., & Zhu, J. (2022). Exploring memorization in adversarial training. In *International conference on learning representations (ICLR)*.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., & Uszkoreit, J. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. In *International conference on learning representations*.

Gontijo-Lopes, R., Smullin, S., Cubuk, E. D., & Dyer, E. (2021). Tradeoffs in data augmentation: An empirical study. In *International conference on learning representations (ICLR)*.

Gowal, S., Qin, C., Uesato, J., Mann, T., & Kohli, P. (2021). Uncovering the limits of adversarial training against norm-bounded adversarial examples. arXiv.

Hataya, R., Zdenek, J., Yoshizoe, K., & Nakayama, H. (2020). Faster AutoAugment: Learning augmentation strategies using backpropagation. In *European conference on computer vision (ECCV)*.

He, K., Zhang, X., Ren, S., & Sun, J. (2016a). *IEEE/CVF conference on computer vision and pattern recognition (CVPR): Deep residual learning for image recognition*.

He, K., Zhang, X., Ren, S., & Sun, J. (2016b). Identity mappings in deep residual networks. In *European conference on computer vision (ECCV)*.

Hendrycks, D., & Dietterich, T. (2019). Benchmarking neural network robustness to common corruptions and perturbations. In *International conference on learning representations (ICLR)*.

Hendrycks*, D., Mu*, N., Cubuk, E.D., Zoph, B., Gilmer, J., & Lakshminarayanan, B. (2020). AugMix: A simple data processing method to improve robustness and uncertainty. In *International conference on learning representations (ICLR)*.

Ho, D., Liang, E., Chen, X., Stoica, I., & Abbeel, P. (2019). Population based augmentation: Efficient learning of augmentation policy schedules. In *International conference on machine learning (ICML)*.

Jia, X., Zhang, Y., Wu, B., Ma, K., Wang, J., & Cao, X. (2022). Las-at: Adversarial training with learnable attack strategy. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 13398–13408).

Kireev, K., Andriushchenko, M., & Flammarion, N. (2022). On the effectiveness of adversarial training against common corruptions. In *Conference on uncertainty in artificial intelligence (UAI)*.

Krizhevsky, A. (2009). In *Learning multiple layers of features from tiny images*. Technical Report.

Kuang, H., Liu, H., Wu, Y., & Ji, R. (2023). Semantically consistent visual representation for adversarial robustness. In *IEEE transactions on information forensics and security*.

Kuang, H., Liu, H., Wu, Y., Satoh, S., & Ji, R. (2023). Improving adversarial robustness via information bottleneck distillation. In *Neural information processing systems (NeurIPS)*.

Li, L., & Spratling, M. (2023a). In *Improved adversarial training through adaptive instance-wise loss smoothing*.

Li, L., & Spratling, M. (2023b). Understanding and combating robust overfitting via input loss landscape analysis and regularization. In *Pattern recognition*.

Li, L., & Spratling, M. W. (2023c). In *International conference on learning representations (ICLR): Data augmentation alone can improve adversarial training*.

Li, L., Wang, Y., Sitawarin, C., & Spratling, M. (2024). OODRobustBench: Benchmarking and analyzing adversarial robustness under distribution shift. In *International conference on machine learning (ICML)*.

Li, Y., Hu, G., Wang, Y., Hospedales, T., Robertson, N. M., & Yang, Y. (2020). Differentiable automatic data augmentation. In *European conference on computer vision (ECCV)*.

Lim, S., Kim, I., Kim, T., Kim, C., & Kim, S. (2019). Fast AutoAugment. In *Neural information processing systems (NeurIPS)*.

Lin, C., Guo, M., Li, C., Yuan, X., Wu, W., Yan, J., & Ouyang, W. (2019). Online hyper-parameter learning for auto-augmentation strategy. In *IEEE/CVF international conference on computer vision (ICCV)*.

Liu, A., Huang, Z., Huang, Z., & Wang, N. (2021). Direct differentiable augmentation search. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 12219–12228).

Liu, H., & Satoh, S. (2023). Rethinking adversarial training with a simple baseline. arXiv preprint arXiv:2306.07613.

Liu, H., Zhong, Z., Sebe, N., & Satoh, S. (2023). Mitigating robust overfitting via self-residual-calibration regularization. *Artificial Intelligence, 317*, 103877.

Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., & Xie, S. (2022). A ConvNet for the 2020s. In *IEEE/CVF conference on computer vision and pattern recognition (CVPR)*.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., & Vladu, A. (2018). Towards deep learning models resistant to adversarial attacks. In *International conference on learning representations (ICLR)*.

Miao, N., Rainforth, T., Mathieu, E., Dubois, Y., Teh, Y.W., Foster, A., & Kim, H. (2023). In *Learning instance-specific augmentations by capturing local invariances*.

Mo, Y., Wu, D., Wang, Y., Guo, Y., & Wang, Y. (2022). When adversarial training meets vision transformers: Recipes from training to architecture. In *Neural information processing systems (NeurIPS)*.

Moosavi-Dezfooli, S.-M., Fawzi, A., Uesato, J., & Frossard, P. (2019). Robustness via curvature regularization, and vice versa. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 9078–9086).

Müller, S. G., & Hutter, F. (2021). Trivialaugment: Tuning-free yet state-of-the-art data augmentation. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 774–782).

Pang, T., Lin, M., Yang, X., Zhu, J., & Yan, S. (2022). Robustness and accuracy could be reconcilable by (proper) definition. In *International conference on machine learning* (pp. 17258–17277).

Qiu, J., Li, L., Sun, J., Peng, J., Shi, P., Zhang, R., & Lo, B. (2023). Large AI models in health informatics: Applications, challenges, and the future. In *IEEE journal of biomedical and health informatics (JBHI)*.

Rade, R., & Moosavi-Dezfooli, S.-M. (2022). Reducing excessive margin to achieve a better accuracy vs. robustness trade-off. In *International conference on learning representations (ICLR)*.

Rebuffi, S.-A., Gowal, S., Calian, D.A., Stimberg, F., Wiles, O., & Mann, T. (2021). Data augmentation can improve robustness. In *Neural information processing systems (NeurIPS)*.

Rice, L., Wong, E., & Kolter, J. Z. (2020). Overfitting in adversarially robust deep learning. In *International conference on machine learning (ICML)*.

Ross, A.S., & Doshi-Velez, F. (2018). Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. In *AAAI conference on artificial intelligence (AAAI)*.

Schwinn, L., Raab, R., Nguyen, A., Zanca, D., & Eskofier, B. (2023). Exploring misclassifications of robust neural networks to enhance adversarial attacks. In *Applied intelligence*.

Sehwag, V., Mahloujifar, S., Handina, T., Dai, S., Xiang, C., Chiang, M., & Mittal, P. (2022). Robust learning meets generative models: Can proxy distributions improve adversarial robustness? *International conference on learning representations (ICLR)*.

Singh, N. D., Croce, F., & Hein, M. (2023). Revisiting adversarial training for ImageNet: Architectures, training and generalization across threat models. In *Neural information processing systems (NeurIPS)*.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., & Fergus, R. (2014). Intriguing properties of neural networks. In *International conference on learning representations (ICLR)*.

Tack, J., Yu, S., Jeong, J., Kim, M., Hwang, S.J., & Shin, J. (2022). Consistency regularization for adversarial robustness. In *AAAI conference on artificial intelligence (AAAI)*.

Wang, H., & Wang, Y. (2022). Self-ensemble adversarial training for improved robustness. In *International conference on learning representations (ICLR)*.

Wang, H., Xiao, C., Kossaifi, J., Yu, Z., Anandkumar, A., & Wang, Z. (2021). AugMax: Adversarial composition of random augmentations for robust training. In *Neural information processing systems (NeurIPS)*.

Wang, Y., Zou, D., Yi, J., Bailey, J., Ma, X., & Gu, Q. (2020). Improving adversarial robustness requires revisiting misclassified examples. In *International conference on learning representations (ICLR)*.

Wang, Z., Pang, T., Du, C., Lin, M., Liu, W., & Yan, S. (2023). Better diffusion models further improve adversarial training. In *International conference on machine learning (ICML)*.

Williams, R.J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. In *Machine learning*.

Wong, E., Rice, L., & Kolter, J. Z. (2020). Fast is better than free: Revisiting adversarial training. In *International conference on learning representations (ICLR)*.

Wu, D., Xia, S.-T., & Wang, Y. (2020). Adversarial weight perturbation helps robust generalization. In *Neural information processing systems (NeurIPS)*.

Yu, C., Han, B., Gong, M., Shen, L., Ge, S., Bo, D., & Liu, T. (2022). Robust weight perturbation for adversarial training. In *International joint conference on artificial intelligence (IJCAI)*.

Yun, S., Han, D., Oh, S.J., Chun, S., Choe, J., & Yoo, Y. (2019). Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 6023–6032).

Zagoruyko, S., & Komodakis, N. (2016). Wide residual networks. In *British machine vision conference (BMVC)*.

Zhang, H., Cisse, M., Dauphin, Y. N., & Lopez-Paz, D. (2017). mixup: Beyond empirical risk minimization. arXiv preprint arXiv:1710.09412.

Zhang, H., Yu, Y., Jiao, J., Xing, E., Ghaoui, L. E., & Jordan, M. (2019). Theoretically principled trade-off between robustness and accuracy. In *International conference on machine learning (ICML)*.

Zhang, X., Wang, Q., Zhang, J., & Zhong, Z. (2020). Adversarial AutoAugment. In *International conference on learning representations (ICLR)*.

Zhong, Z., Zheng, L., Kang, G., Li, S., & Yang, Y. (2020). Random erasing data augmentation. In *AAAI conference on artificial intelligence (AAAI)*.