The Robustness-Generalization Dilemma: Trade-off via Multi-Objective Optimization

Ran Wang[1], Haojie Zhai[2], Wenhui Wu[3], Le Ou-Yang[4], Wing W. Y. Ng[5,*]

[1]School of Artificial Intelligence, Shenzhen University, Shenzhen 518060, China
[2]School of Mathematical Sciences, Shenzhen University, Shenzhen 518060, China
[3]College of Electronic & Information, Engineering, Shenzhen University, Shenzhen 518060, China
[4]Faculty of Engineering, Shenzhen MSU-BIT University, Shenzhen, China
[5]School of Computer Science and Engineering, South China University of Technology, Guangzhou, China
Emails: {wangran@, 2300201002@email, wuwenhui@}szu.edu.cn, leouyang@smbu.edu.cn, wingng@ieee.org

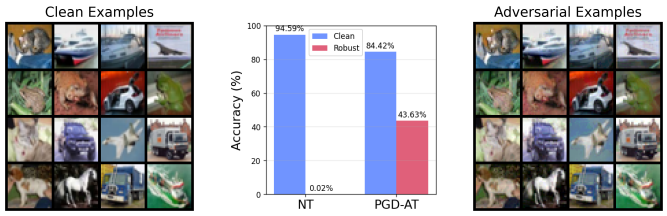
***Abstract*—Existing studies reveal an inherent conflict between the generalization ability and robustness of deep models, known as the robustness-generalization dilemma. Specifically, in the mid-to-late stages of model training, the improvement in robustness is often accompanied by a decrease in generalization ability, and vice versa. This dilemma has been extensively discussed for more than a decade but remains incompletely addressed, indicating the hardness of improving the two performance indicators simultaneously. Existing works usually purse one aspect at the expense of the other, e.g., by optimizing single objectives (such as adversarial loss or clean loss) or a scalar combination of them, causing difficulties to achieve an effective trade-off between them. In this paper, for the first time, we introduce the multi-objective optimization (MOO) technique to deal with the robustness-generalization dilemma, and propose the Robustness-Generalization Trade-off framework based on MOO (RGTro-MOO). The proposed RGTro-MOO is incorporated into adversarial training, which dynamically and stage-wisely optimizes the attack strategies to trade-off robustness and generalization. Non-dominated sorting genetic algorithm II is used to evolve and select the Pareto-optimal solutions, and Pareto front is visualized to intuitively show the dilemma. Experiments on benchmark image classification datasets validate the feasibility of applying MOO to trade-off robustness and generalization, and comparisons with some state-of-the-art methods demonstrate its superior performance.**

***Index Terms*—Robustness, Generalization, Multi-objective Optimization, Trade-off**

I. Introduction

Robustness and *generalization* are two fundamental performance indicators in modern deep learning, which characterize how well a model performs on unseen clean data and perturbed data, respectively. Especially, adversarial robustness measures the model's resilience against delicately crafted perturbations, reflecting the model's reliability under malicious conditions [1]. As deep learning is increasingly applied to domains with high security requirements, achieving both good generalization and strong robustness has become a critical yet challenging research problem.

This work was supported in part by the National Natural Science Foundation of China (Grants 62576214, 62476100 and U24A20322); in part by the Guangdong Basic and Applied Basic Research Foundation (Grant 2024B1515020109). (* Corresponding author.)



(a) Clean (b) Accuracy (c) Adversarial

Fig. 1. Robustness-Generalization Dilemma (the robust accuracy is evaluated by PGD-10).

Prior research reveals that robustness and generalization often exhibit some conflicts. For example, as the most effective adversarial defense methods, adversarial training (AT) and its variants [2]–[4] tend to significantly improve model robustness but at the cost of reduced clean accuracy [5], [6]. Conversely, models trained by maximizing generalization on clean data often fail catastrophically under adversarial perturbations. This conflict is known as the robustness-generalization dilemma. Fig. 1(a) and Fig. 1(c) respectively demonstrate some clean examples and their corresponding adversarial examples in dataset CIFAR-10. Fig. 1(b) further reports the clean accuracy and robust accuracy of two models derived by natural training (NT) and AT, respectively. Obviously, some conflicts can be observed from Fig. 1(b), i.e., compared with the NT-model, the AT-model exhibits much better robustness (0.02%→43.63%) but worse clean accuracy (94.59%→84.42%). Existing methods typically focus on optimizing one aspect while overlooking the other; the trade-off between them has not been systematically investigated, especially across different training stages.

Multi-objective optimization (MOO), as a classic tool for optimizing two or more conflicting objectives, offers a principled framework by searching for the Pareto-optimal solutions [7]. However, to the best of our knowledge, applying MOO to trade-off robustness and generalization for AT has not been investigated yet. Motivated by this, in this paper, we propose a **R**obustness-**G**eneralization **Tr**ade-**o**ff framework based on **MOO** (RGTro-MOO). Specifically, we formulate robustness and generalization as distinct optimization objectives,

one is measured through adversarial loss under parameterized attacks, while the other is captured via clean loss. A multi-objective search space is constructed for parameterized attack strategy, and the non-dominated sorting genetic algorithm II (NSGA-II) is used to approximate the Pareto front, thereby revealing the inherent conflict and balancing the two objectives. The contributions of this work are summarized as follows:

- We introduce the RGTro-MOO framework, which dynamically trade-off the robustness and generalization across different training stages, and provides new insights into their inherent conflict by Pareto front analysis.
- We propose a stage-wise scheme for attack strategy optimization. NSGA-II is used to explore the parameter space and find the Pareto-optimal solutions, overcoming the limitation of single-objective training.
- We compare our method with state-of-the-art methods on benchmark image classification datasets. Comparison results demonstrate that it can achieve superior balance between robustness and generalization.

II. Related Work

A. Robustness-Generalization Dilemma

Recent studies have introduced a series of adversarial attack methods, such as fast gradient sign method (FGSM) [1], projected gradient descent method (PGD) [2], Carlini & Wagner attack (CW) [8], auto attack (AA) [9], etc., drawing significant attention to the vulnerability of deep models. Subsequently, many robust learning methods have been proposed to promote the adversarial robustness of models. AT is a canonical approach [2] by formulating the training process as a min-max optimization problem:

$$\min_{\theta} \mathbb{E}_{(\mathbf{x},y)\sim D} \max_{\|\delta\|_p<\epsilon} \ell(\mathbf{x}+\delta, y; \theta), \tag{1}$$

where $(\mathbf{x}, y)$ is a training example, D is the training distribution, δ is the adversarial perturbation, ϵ is the upper bound limit for δ, θ represents network parameters, and ℓ is the loss function. Given the training set $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$, problem (1) has the following surrogate form:

$$\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} \ell(\mathbf{x}_{i,\text{adv}}^{\mathbf{a}^+}, y; \theta), \tag{2}$$

where $\mathbf{x}_{i,\text{adv}}^{\mathbf{a}^+}$ represents the perturbed $\mathbf{x}_i$ by the standard PGD attack $\mathbf{a}^+ = [\frac{8}{255}, \frac{2}{255}, 10]$, i.e., PGD attack with attack strength $\frac{8}{255}$, step size $\frac{8}{255}$, and number of steps 10.

AT can effectively enhance model robustness under various attacks. However, both empirically and theoretically, researchers have revealed that it may significantly reduce the generalization ability on clean data [5], [6]. Although there exist many techniques to improve *generalization* or *robustness* (e.g., regularization, data augmentation, instance reweighting [10], [11], etc.), the tension between them remains an unsolved problem. Some works aim to mitigate the conflict between robustness and accuracy [12]–[14]. For example, TRADES [5] balances the two objectives by introducing a KL divergence term between clean and perturbed examples, MART [15] further emphasizes the misclassified examples and designs a re-weighting scheme, and GAIRAT leverages geometric information for adaptive instance weighting [16]. However, none of them has formally modeled this conflict, thereby limiting the subsequent analysis. Besides, it is noteworthy that different stages of AT may be suitable with different attack strategies, which has been overlooked by the existing AT methods.

B. Multi-objective Optimization (MOO)

MOO offers a principled framework to handle conflicting goals by searching for the Pareto-optimal solutions rather than a single optimal solution. Given k conflicting objectives $f_1, \ldots, f_k$, MOO is mathematically formulated as:

$$\begin{aligned} \min_{\mathbf{a}} \quad & \mathcal{F}(\mathbf{a}) = (f_1(\mathbf{a}), f_2(\mathbf{a}), \ldots, f_k(\mathbf{a}))^\top \\ \text{s.t.} \quad & \mathbf{a} \in \Omega \end{aligned}, \tag{3}$$

where $\mathcal{F}$ represents the multiple objectives, $\mathbf{a}$ is a solution, and Ω is the solution space. The goal of MOO is to find a set of Pareto-optimal solutions, which are defined as follows.

Definition 1: $\mathbf{a}^1$ is said to Pareto dominate $\mathbf{a}^2$ if $f_i(\mathbf{a}^1) \leq f_i(\mathbf{a}^2)$ for every $i \in \{1, \ldots, k\}$ and $f_j(\mathbf{a}^1) < f_j(\mathbf{a}^2)$ for at least one index $j \in \{1, \ldots, k\}$, denoted as $\mathbf{a}^1 \preceq \mathbf{a}^2$.

Definition 2: A solution $\mathbf{a}^*$ is said to be Pareto-optimal if there is no other solution $\mathbf{a}$ such that $\mathbf{a} \preceq \mathbf{a}^*$.

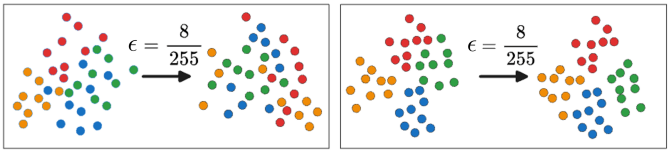
Definition 3: The set of all Pareto-optimal solutions $\mathbf{a}$ is called Pareto-optimal set. The set of all Pareto-optimal vectors $\mathcal{F}(\mathbf{a})$ is called Pareto front.

Multi-objective evolutionary algorithms (MOEAs) are classic methods for solving MOO, which adopt population-driven heuristic algorithms for optimizing the multiple objectives. They have been applied to many machine learning problems [17], such as neural architecture search [18], resource-accuracy trade-off [19], hyperparameter tuning [20], etc. NSGA-II [7], as a foundational work, provides an efficient mechanism for searching the Pareto-optimal solutions, demonstrating the usefulness of Pareto fronts for achieving trade-offs between competing metrics.

However, to the best of our knowledge, such methods have not been applied to deal with the robustness-generalization dilemma so far. On the one hand, most prior defense methods optimize a single objective or a scalar combination of single objectives [5], none of them treat robustness and generalization as multiple conflicting objectives. On the other hand, robust overfitting has been observed [6], suggesting that AT with static attack strategy $\mathbf{a}^+ = [\frac{8}{255}, \frac{2}{255}, 10]$ is not ideal; treating robustness and generalization as multiple conflicting objectives and exploring suitable attack strategy stage-wisely can make the training process more rational.

III. Methodology

The effectiveness of AT is largely dependent on the quality of the generated adversarial examples, which is determined by the attack strategy adopted. Static strategy is unable to dynamically capture the training status of the model, thereby has difficulties to achieve trade-off between *robustness* and



(a) Early Training Stage (b) Mid-to-Late Training Stages

Fig. 2. Limitations of static attack strategy.

generalization adaptively. As shown in Fig. 2, a static strategy maybe too strong for the early training stage such that the model's generalization ability is destroyed catastrophically; while it maybe too weak for the mid-to-late training stages such that the robustness cannot be improved further.

Problem Formulation: Consider a network model F parameterized by θ, i.e., F_θ. Given a set of clean training examples $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$, our goal is to dynamically and stage-wisely optimize attack strategy $\mathbf{a}$ by applying MOO technique, where the optimized attack strategy $\mathbf{a}^*$ can enable effective updates of model F_θ such that the updated model can perform well regarding two probably conflicting indicators: clean accuracy and robust accuracy.

Attack Strategy Parameterization: Following the mainstream AT methods, we adopt PGD attack [2] for generating adversarial examples during training:

$$\mathbf{x}_i^{(t+1)} = \prod_{B_\epsilon(\mathbf{x}_i)} \left(\mathbf{x}_i^{(t)} + \eta \cdot \operatorname{sign}\left(\nabla_{\mathbf{x}_i} \ell(F_\theta(\mathbf{x}_i^{(t)}), y_i)\right)\right), \quad (4)$$

where $t = 1, \ldots, I$ is the index of attacking step, I is the total number of steps, η is the step size, ϵ is the perturbation upper bound, and $\prod_{B_\epsilon(\mathbf{x}_i)}(\cdot)$ denotes projecting the perturbed example onto a ℓ_∞-ball of radius ϵ and center $\mathbf{x}_i$ when the perturbation size exceeds the upper bound. In this case, the attack strategy can be fully parameterized, i.e.,

$$\mathbf{a} = [\epsilon, \eta, I], \quad (5)$$

where $\mathbf{a} \in \mathcal{A}$, and $\mathcal{A} = [\epsilon_{\min}, \epsilon_{\max}] \times [\eta_{\min}, \eta_{\max}] \times [I_{\min}, I_{\max}]$ denotes the search space.

Virtual Model Update: Note that performing candidate attacks on the whole training set is too computationally expensive. Thus, we first sample a subset of clean examples $\mathcal{D}_{eval} \subset \mathcal{D}$, then apply attack $\mathbf{a}$ to each $\mathbf{x}_i \in \mathcal{D}_{eval}$ to generate adversarial example $\mathbf{x}_{i,\text{adv}}^{\mathbf{a}}$. The model F_θ is virtually updated once via back-propagation using these adversarial examples, yielding an updated model $F_{\theta'}$. The updating rule can be expressed as a single-step gradient descent:

$$\theta' = \theta - \frac{\alpha}{|\mathcal{D}_{eval}|} \nabla_\theta \sum_{(\mathbf{x}_i, y_i) \in \mathcal{D}_{eval}} \ell\left(F_\theta(\mathbf{x}_{i,\text{adv}}^{\mathbf{a}}), y_i\right), \quad (6)$$

where α is the learning rate and $\ell(\cdot, \cdot)$ is the classification loss function (e.g., cross-entropy loss).

Evaluation Objectives: After generating adversarial examples and performing a one-step virtual update, the updated model $F_{\theta'}$ is evaluated using two complementary objectives [21]: generalization loss and adversarial loss. Specifically, generalization loss is the loss of $F_{\theta'}$ on clean examples, denoted as $\mathcal{L}_{\text{gen}}(\mathbf{a})$, and adversarial loss is the loss of $F_{\theta'}$ on perturbed examples under standard PGD attack $\mathbf{a}^+ = [\frac{8}{255}, \frac{2}{255}, 10]$, denoted as $\mathcal{L}_{\text{rob}}(\mathbf{a})$, i.e.,

$$\begin{cases} \mathcal{L}_{\text{gen}}(\mathbf{a}) = \frac{1}{|\mathcal{D}_{eval}|} \sum_{(\mathbf{x}_i, y_i) \in \mathcal{D}_{eval}} \ell\left(F_{\theta'}(\mathbf{x}_i), y_i\right) \\ \mathcal{L}_{\text{rob}}(\mathbf{a}) = \frac{1}{|\mathcal{D}_{eval}|} \sum_{(\mathbf{x}_i, y_i) \in \mathcal{D}_{eval}} \ell\left(F_{\theta'}(\mathbf{x}_{i,\text{adv}}^{\mathbf{a}^+}), y_i\right) \end{cases}, \quad (7)$$

where $\mathbf{x}_{i,\text{adv}}^{+}$ is the perturbed $\mathbf{x}_i$ by standard PGD attack $\mathbf{a}^+$.

MOO Formulation: The subsequent training aims to reduce both of the two loss terms in Eq. (7) simultaneously. However, as aforementioned, these two objectives may be inherently conflicting with each other. Especially, in the mid-to-late training stages, improvement in one objective usually causes decline in the other. This naturally leads to a *bi-objective optimization problem*:

$$\begin{aligned} \min_{\mathbf{a}} \quad & \mathcal{L}(\mathbf{a}) = \left(\mathcal{L}_{\text{gen}}(\mathbf{a}), \mathcal{L}_{\text{rob}}(\mathbf{a})\right)^\top \\ \text{s.t.} \quad & \mathbf{a} \in \mathcal{A} \end{aligned}. \quad (8)$$

An ideal attack strategy $\mathbf{a}^*$ should yield low values for both $\mathcal{L}_{\text{gen}}$ and $\mathcal{L}_{\text{rob}}$, which can be searched by MOEAs.

Overall Workflow of RGTro-MOO: We integrate the attack strategy search, the virtual model update and the MOO paradigm to propose the RGTro-MOO framework, which is briefly summarized in Fig. 3, and is detailed in Algorithm 1. Note that in step 8, NSGA-II is applied for selecting the non-dominated solutions. Besides, in step 12, different principals can be designed to select the preferred solution as the output:

- Robust point: $\mathbf{a}^*$ is selected as the solution with the best robustness, i.e.,

$$\mathbf{a}^* = \operatorname*{argmin}_{\mathbf{a} \in \mathcal{A}} \mathcal{L}_{\text{rob}}(\mathbf{a}). \quad (9)$$

Algorithm 1: RGTro-MOO

Input: Model F_θ, evaluation subset $\mathcal{D}_{eval} \subset \mathcal{D}$, parameter space $\mathcal{A}$, standard attack $\mathbf{a}^+$, population size P, number of generations G.
Output: Optimized attack $\mathbf{a}^*$.

1: Initialize random population $\mathbb{P} = \{\mathbf{a}_1, \ldots, \mathbf{a}_P\} \subset \mathcal{A}$;
2: **for** $g = 1$ to G **do**
3: **for** each candidate $\mathbf{a} \in \mathbb{P}$ **do**
4: Generate adversarial example $\mathbf{x}_{i,\text{adv}}^{\mathbf{a}}$ for each $\mathbf{x}_i \in \mathcal{D}_{eval}$;
5: Perform one-step virtual update via (6) to obtain $F_{\theta'}$;
6: Evaluate objectives $\mathcal{L}_{\text{gen}}(\mathbf{a})$ and $\mathcal{L}_{\text{rob}}(\mathbf{a})$ via (7);
7: **end for**
8: Select non-dominated solutions using Pareto dominance;
9: Apply crossover and mutation to generate new candidates;
10: **end for**
11: Obtain the approximate Pareto front $\mathcal{P}$;
12: Select solution $\mathbf{a}^* \in \mathcal{P}$ based on stage-specific preference;
13: **return** $\mathbf{a}^*$.

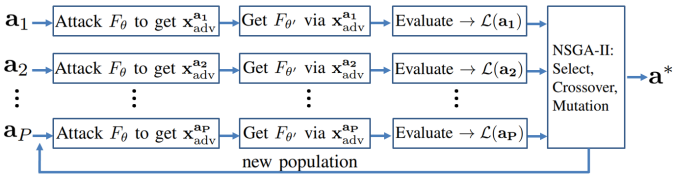


Fig. 3. Framework of RGTro-MOO.

TABLE I
COMPARISON RESULTS (% OF ACCURACY) ON DATASET CIFAR-10.

Method	Clean	FGSM	PGD-10	PGD-20	PGD-50	C&W	AA	Avg±Std	Time
PGD-AT	82.12	56.61	51.17	50.28	50.00	24.65	46.63	46.56±11.21	43857s
TRADES	83.42	**58.77**	52.68	51.73	51.49	30.24	**48.65**	53.85±15.78	44057s
MART	82.00	56.92	49.76	48.30	47.83	22.52	43.36	50.10±17.70	29750s
FAT	**86.98**	53.99	43.94	41.87	41.28	08.96	39.55	45.22±23.08	45154s
AT+RiFT	76.78	44.09	38.57	37.77	37.45	23.13	35.19	41.85±16.67	127665s
MAIL	84.03	55.96	51.29	50.64	50.31	24.19	46.82	51.89±17.54	101026s
TRA^{2}SO	79.06	56.78	51.94	50.85	50.51	29.42	47.25	52.26±14.67	47694s
PGD-AT-linear (8/255)	89.35	51.19	37.58	35.08	34.41	01.67	32.23	32.03±16.34	——
PGD-AT-linear (12/255)	86.58	55.37	44.68	42.65	41.92	08.66	38.52	38.63±15.76	——
PGD-AT-linear (16/255)	84.80	56.78	47.13	45.18	44.48	15.78	40.05	41.57±13.79	——
AT-RGTro-MOO (Robust)	80.51	56.89	53.73	52.04	51.72	**38.29**	47.12	**54.33±13.00**	66489s
AT-RGTro-MOO (Ideal)	81.14	56.96	53.32	51.30	50.90	30.80	46.98	53.06±13.86	63673s
AT-RGTro-MOO (Balanced)	80.64	57.46	**53.89**	**52.27**	**52.01**	35.21	47.15	54.09±12.69	72580s

- Ideal point: $\mathbf{a}^*$ is selected as the solution with the smallest distance to the optimal point, i.e.,

$$\mathbf{a}^* = \underset{\mathbf{a}\in\mathcal{A}}{\operatorname{argmin}} \operatorname{Dist}(\mathbf{a}, \hat{\mathbf{a}}), \tag{10}$$

  where $\hat{\mathbf{a}} = [\min_{\mathbf{a}} \mathcal{L}_{\text{gen}}(\mathbf{a}), \min_{\mathbf{a}} \mathcal{L}_{\text{rob}}(\mathbf{a})]$, and Dist can be any distance metric, such as Euclidean distance, Cosine distance, Gaussian distance, etc.
- Balanced point: $\mathbf{a}^*$ is selected as the solution with the smallest weighted sum of generalization loss and adversarial loss, i.e.,

$$\mathbf{a}^* = \underset{\mathbf{a}\in\mathcal{A}}{\operatorname{argmin}} \left[\alpha\mathcal{L}_{\text{gen}}(\mathbf{a}) + \beta\mathcal{L}_{\text{rob}}(\mathbf{a})\right], \tag{11}$$

  where α and β are coefficient parameters, both taking values in $[0, 1]$ and $\alpha + \beta = 1$. For convenience, we can simply set $\alpha = \beta = 0.5$.

AT with RGTro-MOO: Finally, RGTro-MOO is integrated into AT to explore a suitable attack strategy stage-wisely, which can benefit both objectives. To reduce time cost, RGTro-MOO is triggered every K training epochs rather than each epoch, which also provides convenience for monitoring the status of different training stages. The training process, named AT-RGTro-MOO, is described in Algorithm 2.

Algorithm 2: AT-RGTro-MOO

Input: Training set $\mathcal{D}$, training epoch T, batch size B, optimization frequency K, attack parameter space $\mathcal{A}$, standard attack $\mathbf{a}^+$, population size P, number of generations G.
Output: Model F_θ.

```
Randomly initialize model parameters θ;
Sample an evaluation subset D_eval ⊂ D;
for epoch = 1 : T do
    if epoch mod K == 0 then
        a* =RGTro-MOO(F_θ, D_eval, A, a+, P, G);
    end if
    for 1 : num_batch do
        Sample a new size-B mini-batch;
        Generate x_{i,adv}^{a*} for each x_i in the mini-batch;
        Update θ via gradient descent based on the batch of x_i
        and x_{i,adv}^{a*};
    end for
end for
return The final F_θ.
```

IV. EXPERIMENTS

We conduct experiments on two benchmark image classification datasets, i.e., CIFAR-10 and CIFAR-100, where ResNet-18 [22] is adopted as the backbone architecture.

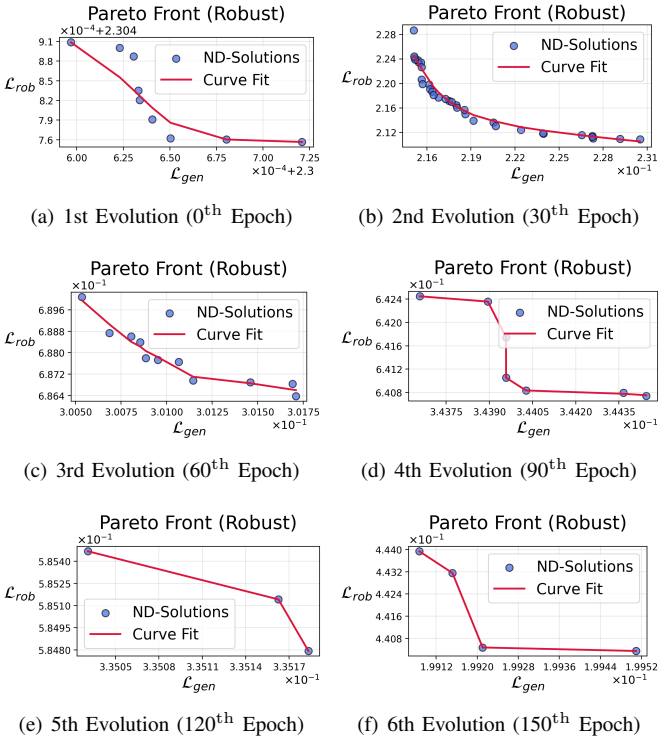
Training Configurations: For the proposed AT-RGTro-MOO, stochastic gradient descent (SGD) is used as the optimizer, with the number of training epochs set to 200, an initial learning rate of 0.1, a momentum of 0.9, and a weight decay coefficient of 5e-4. NSGA-II is implemented using the Geatpy library [23], where the population size P is 30, the number of evolution generations G is 20, the strategy space is set to $\mathcal{A} = [2/255, 16/255] \times [1/255, 4/255] \times [4, 20]$, the crossover and mutation probabilities are 1 and $\frac{1}{3}$, respectively. The evaluation dataset $\mathcal{D}_{eval}$ is randomly sampled from the training set just once at the start of the run with $|\mathcal{D}_{eval}| = 200$, and the alternating optimization frequency K is set to 30.

Evaluation Protocols: To comprehensively evaluate the model's generalization and robustness, we select three types of attack methods: gradient-based FGSM and PGD with l_∞-norm constraints, optimization-based C&W with l_2-norm constraint, and ensemble AA with l_∞-norm constraints. The attack strength is consistently set to $8/255$. For PGD, the step size is set to $2/255$, with 10, 20, and 50 steps respectively; for C&W, the initial constant, confidence, number of iterations, and learning rate are set to 1, 0, 50, and 0.01. FGSM, PGD, and C&W are implemented using the torchattacks library, while AA is implemented using the source code provided in the original paper, selecting its standard mode.

Comparison Results: We compare the proposed AT-RGTro-MOO with several state-of-the-art AT methods, including standard PGD-AT [2], TRADES [5], MART [15], FAT [24], GAIRAT [16], MAIL [25], AT+RiFT [12], and TRA^{2}SO [26]. Besides, we also compare AT-RGTro-MOO with several linearly scheduled attack strategies (PGD-AT-linear), i.e., $\epsilon_{\text{epoch}} = \text{epoch} \times \epsilon_{\max}/200$, where $\epsilon_{\max}$ is set to 8/255, 12/255 and 16/255, respectively. For all the compared methods, PGD-10 is used consistently to perturb examples; for PGD-AT, PGD-AT-linear and AT-RGTro-MOO, we select the best model via PGD-10; while all the other configurations for the compared methods are set based on recommendations in the original papers. The comparison results on CIFAR-10

TABLE II
COMPARISON RESULTS (% OF ACCURACY) ON DATASET CIFAR-100.

Method	Clean	FGSM	PGD-10	PGD-20	PGD-50	C&W	AA	Avg±Std	Time
PGD-AT	57.04	29.63	26.32	25.75	25.52	**22.84**	10.24	23.38±06.79	44098s
TRADES	55.50	30.56	27.82	27.48	27.27	14.37	23.54	29.51±12.61	40519s
MART	53.17	29.18	26.52	25.92	25.84	09.26	22.60	27.50±13.07	29877s
FAT	61.29	25.60	19.73	18.77	18.52	01.94	17.75	23.37±18.22	41485s
GAIRAT	55.23	24.20	20.12	19.30	19.08	09.65	16.36	23.42±14.71	42616s
AT+RiFT	45.08	20.94	18.77	18.54	18.41	17.26	15.38	22.05±10.29	125939s
TRA^{2}SO	51.41	27.61	24.42	23.82	23.58	06.52	21.72	25.58±13.28	41827s
PGD-AT-linear (8/255)	**64.63**	22.39	15.31	14.05	13.78	00.55	12.76	13.14±07.07	——
PGD-AT-linear (12/255)	61.37	25.12	19.21	18.24	17.80	01.30	16.96	16.44±07.97	——
PGD-AT-linear (16/255)	58.94	25.25	20.16	19.40	19.15	01.97	18.02	17.33±07.93	——
AT-RGTro-MOO (Robust)	52.35	30.90	29.59	28.79	28.69	15.64	24.41	30.05±11.11	69391s
AT-RGTro-MOO (Ideal)	50.59	30.99	29.66	**29.09**	28.87	16.82	**24.77**	**30.11±10.23**	65124s
AT-RGTro-MOO (Balanced)	50.45	**31.29**	**29.86**	29.03	**28.89**	16.71	24.56	**30.11±10.23**	65990s

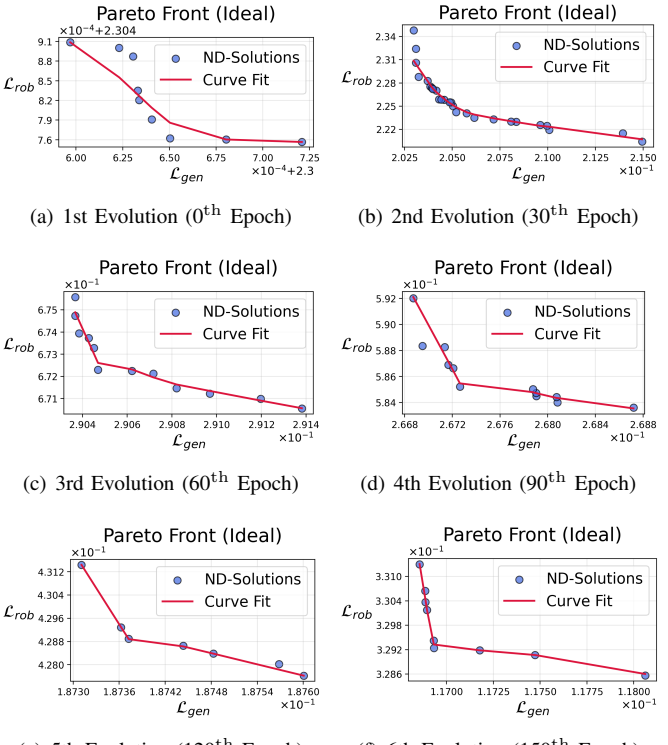


(a) 1st Evolution (0^{th} Epoch) (b) 2nd Evolution (30^{th} Epoch)

(c) 3rd Evolution (60^{th} Epoch) (d) 4th Evolution (90^{th} Epoch)

(e) 5th Evolution (120^{th} Epoch) (f) 6th Evolution (150^{th} Epoch)

Fig. 4. Visualization of Pareto fronts for dataset CIFAR-10 (selecting robust $\mathbf{a}^*$ via Eq. (9)).



(a) 1st Evolution (0^{th} Epoch) (b) 2nd Evolution (30^{th} Epoch)

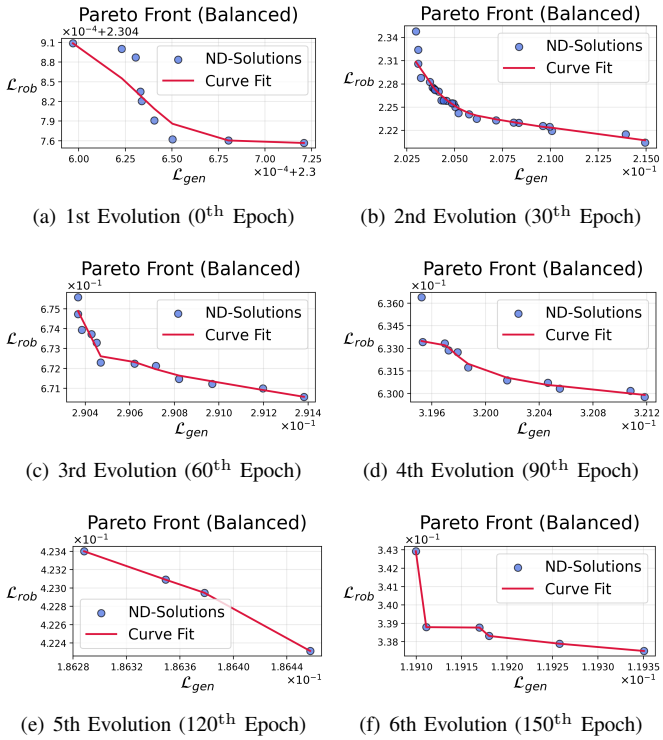
(c) 3rd Evolution (60^{th} Epoch) (d) 4th Evolution (90^{th} Epoch)

(e) 5th Evolution (120^{th} Epoch) (f) 6th Evolution (150^{th} Epoch)

Fig. 5. Visualization of Pareto fronts for dataset CIFAR-10 (selecting ideal $\mathbf{a}^*$ via Eq. (10)).

and CIFAR-100 are reported in Tables I and II, respectively. It can be observed that under most of the chosen attacks, the proposed AT-RGTro-MOO achieves very competitive robust accuracy. At the same time, the sacrifice on clean accuracy is acceptable. The second-last column of Tables I and II reports the average and standard deviation of all the clean and robust accuracies for each method. From the perspective of robustness-generalization trade-off, a higher average value and a lower standard deviation are preferred. Obviously, the proposed AT-RGTro-MOO demonstrates the best trade-off effect among the compared methods. Finally, the last column of Tables I and II reports the training time, demonstrating that the additional overhead of AT-RGTro-MOO is acceptable.

Pareto Front Visualization: With 200 training epochs and optimization frequency 30, the attack strategy evolves 7 times in total, i.e., the 0^{th}, 30^{th}, 60^{th}, 90^{th}, 120^{th}, 150^{th}, and 180^{th} epochs. We record the final objective values (i.e., $\mathcal{L}_{\text{gen}}$ and $\mathcal{L}_{\text{rob}}$) of each solution in the population after the evolution is finished for dataset CIFAR-10, and plot the Pareto fronts as shown in Figs. 4∼6. Clearly and intuitively, Figs. 4∼6 illustrate the inherent conflicts between *robustness* and *generalization*, i.e., there exists a Pareto-optimal set where the solutions cannot be effectively measured by any single objective. Besides, the Pareto front exhibits different characteristics in different training stages, showing that it is necessary to dynamically select attack strategies for different training stages. Fig. 7 further demonstrates the change of clean and robust testing accuracies under PGD-10 attack during the entire training. It can be seen that an effective trade-off is

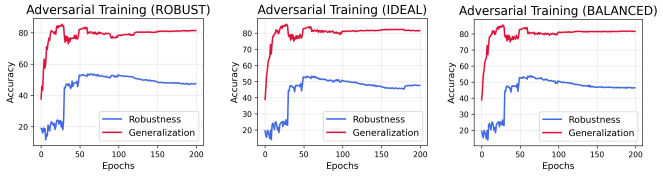


(a) 1st Evolution (0^{th} Epoch) (b) 2nd Evolution (30^{th} Epoch)

(c) 3rd Evolution (60^{th} Epoch) (d) 4th Evolution (90^{th} Epoch)

(e) 5th Evolution (120^{th} Epoch) (f) 6th Evolution (150^{th} Epoch)

Fig. 6. Visualization of Pareto fronts for dataset CIFAR-10 (selecting balanced $\mathbf{a}^*$ via Eq. (11)).



(a) Selecting robust $\mathbf{a}^*$ via Eq. (9) (b) Selecting idea $\mathbf{a}^*$ via Eq. (10) (c) Selecting balanced $\mathbf{a}^*$ via Eq. (11)

Fig. 7. Change of clean testing accuracy and robust testing accuracy under PGD-10 during adversarial training.

realized no matter which principal is adopted for selecting $\mathbf{a}^*$.

V. Conclusion and Future Work

In this work, we investigate the inherent trade-off between *robustness* and *generalization* of deep models using MOO, which jointly quantifies these two competing objectives through adversarial loss and clean loss, and RGTro-MOO framework is proposed to dynamically optimize attack strategies. Our results revealed several key findings: 1) robustness and generalization are fundamentally conflicting objectives, as evidenced by the Pareto front; 2) single-objective optimization is insufficient to achieve a favorable balance; 3) the proposed RGTro-MOO framework consistently outperforms Vanilla AT, demonstrating its effectiveness in balancing the two objectives. These results highlight the necessity of treating robustness and generalization as a multi-objective problem rather than pursuing one at the expense of the other. In the future, we may consider to try more advanced MOO techniques rather than NSGA-II, and figure out a reasonable selection operator to choose solutions from the Pareto front.

References

[1] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[2] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *ICLR*, 2018.

[3] A. Shafahi, M. Najibi, G. Ghiasi, Z. Xu, J. Dickerson, C. Studer, L. Davis, G. Taylor, and T. Goldstein, "Adversarial training for free!," in *NeurIPS*, 2019.

[4] E. Wong, L. Rice, and J. Z. Kolter, "Fast is better than free: Revisiting adversarial training," in *ICLR*, 2020.

[5] H. Zhang, Y. Yu, Q. Jiao, E. P. Xing, L. E. Ghaoui, and M. I. Jordan, "Theoretically principled trade-off between robustness and accuracy," in *NeurIPS*, 2019.

[6] L. Rice, E. Wong, and J. Z. Kolter, "Overfitting in adversarially robust deep learning," *Proceedings of Machine Learning Research*, vol. 119, pp. 8092–8104, 2020.

[7] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," in *IEEE Transactions on Evolutionary Computation*, vol. 6, pp. 182–197, 2002.

[8] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *IEEE Symposium on Security and Privacy*, pp. 39–57, 2017.

[9] F. Croce and M. Hein, "Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks," in *ICML*, 2020.

[10] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014.

[11] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," in *ICLR*, 2018.

[12] K. Zhu, X. Hu, J. Wang, X. Xie, and G. Yang, "Improving generalization of adversarial training via robust critical fine-tuning," in *ICCV*, pp. 4424–4434, 2023.

[13] K. Li, J.-C. N. Ferrand, R. Sheatsley, B. Hoak, Y. Beugin, E. Pauley, and P. McDaniel, "On the robustness tradeoff in fine-tuning," *arXiv preprint arXiv:2503.14836*, 2025.

[14] W. Lin and L. Liao, "Towards sustainable adversarial training with successive perturbation generation," *Frontiers of Information Technology and Electronic Engineering*, vol. 25, no. 4, pp. 527–539, 2024.

[15] Y. Wang, Y. Yu, S. Kumar, K. Sapra, and S.-F. Liu, "MART: Misclassification aware adversarial training," in *ICLR*, 2019.

[16] H. Zhang, J. Zhu, Y. Yu, and M. I. Jordan, "GAIRAT: Geometry-aware instance-reweighted adversarial training," in *NeurIPS*, 2020.

[17] W. Chen, X. Zhang, B. Lin, X. Lin, H. Zhao, Q. Zhang, and J. T. Kwok, "Gradient-based multi-objective deep learning: Algorithms, theories, applications, and beyond," *arXiv preprint arXiv:2501.10945*, 2025.

[18] T. Elsken, J. H. Metzen, and F. Hutter, "Neural architecture search: A survey," *Journal of Machine Learning Research*, vol. 20, no. 55, pp. 1–21, 2019.

[19] M. Tan and Q. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *ICML*, 2019.

[20] Z. Lu *et al.*, "Multi-objective neural architecture search: A survey," *ACM Computing Surveys*, vol. 54, no. 8, pp. 1–34, 2021.

[21] X. Jia, Y. Zhang, B. Wu, K. Ma, J. Wang, and X. Cao, "LAS-AT: Adversarial training with learnable attack strategy," in *CVPR*, pp. 13388–13398, 2022.

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, pp. 770–778, 2016.

[23] J. Jazzbin, "Geatpy: the genetic and evolutionary algorithm toolbox with high performance in python." https://github.com/geatpy-dev/geatpy/, 2020.

[24] J. Zhang, X. Xu, B. Han, G. Niu, L. Cui, M. Sugiyama, and M. Kankanhalli, "Attacks which do not kill training make adversarial learning stronger," in *ICML*, 2020.

[25] Q. Wang, F. Liu, B. Han, T. Liu, C. Gong, G. Niu, M. Zhou, and M. Sugiyama, "Probabilistic margins for instance reweighting in adversarial training," in *NeurIPS*, vol. 34, pp. 23258–23269, 2021.

[26] H. Zhai, R. Wang, W. Wu, and Y. Jia, "Trade-off learning for robustness and generalization based on adaptive strategy optimization," *Journal of Software*, vol. 37, no. 4, pp. 1472–1491, 2026.