International Journal of Computer Vision (2023) 131:2682–2698
https://doi.org/10.1007/s11263-023-01826-6

MANUSCRIPT

Anti-Bandit for Neural Architecture Search

Runqi Wang[1] · Linlin Yang[3] · Hanlin Chen[3] · Wei Wang[1] · David Doermann[4] · Baochang Zhang[1,2,5]

Received: 1 September 2022 / Accepted: 15 May 2023 / Published online: 17 June 2023

Abstract
Neural Architecture Search (NAS) is a highly challenging task that requires consideration of search space, search efficiency, and adversarial robustness of the network. In this paper, to accelerate the training speed, we reformulate NAS as a multi-armed bandit problem and present Anti-Bandit NAS (ABanditNAS) method, which exploits Upper Confidence Bounds (UCB) to abandon arms for search efficiency and Lower Confidence Bounds (LCB) for fair competition between arms. Based on the presented ABanditNAS, the adversarially robust optimization and architecture search can be solved in a unified framework. Specifically, our proposed framework defends against adversarial attacks based on a comprehensive search of denoising blocks, weight-free operations, Gabor filters, and convolutions. The theoretical analysis on the rationality of the two confidence bounds in ABanditNAS are provided and extensive experiments on three benchmarks are conducted. The results demonstrate that the presented ABanditNAS achieves competitive accuracy at a reduced search cost compared to prior methods.

Keywords NAS · Bandit · Adversarial defense

1 Introduction

Deep Convolutional Neural Networks (DCNNs) have dominated as the best performers on various computer vision tasks such as image classification He et al. (2016), instance segmentation Long et al. (2015) and object detection Szegedy et al. (2015) due to the great success of deep network architecture design. With the increasing demand for architecture engineering, instead of designing complex architectures manually, Neural Architecture Search (NAS) has been proved as one of the best approaches for many tasks by generating delicate neural architectures. However, most existing NAS methods neglect the adversarial robustness of the network and suffer from low search efficiency.

The adversarial robustness of a network is critical as most existing deep models are susceptible to adversarial attacks Liu et al. (2016); Carlini et al. (2017); Goodfellow et al. (2015); Szegedy et al. (2013), *i.e.* an imperceptible perturbation to input images can cause the models to perform incorrectly. Especially, Szegedy *et al.* Szegedy et al. (2013) observe that these perturbations are transferable, which means the perturbations generated for one model might mislead other models as well. This highlights the importance of a network's adversarial robustness in real-world scenarios. While many methods have been proposed to defend against the attacks, improving the network training process is one of the most common strategy. These methods inject adversarial examples into the training data to retrain the network Goodfellow et al. (2015); Kurakin et al. (2016); Athalye et al. (2018). Similarly, pre-processing defense methods modify adversarial inputs to resemble clean inputs Samangouei et al. (2018); Liao et al. (2018) by transforming adversarial images to clean images before feeding into the classifier. However, finding adversarially robust architectures using NAS shows even more promising results Cubuk et al. (2017); Kotyan et al. (2020); Guo et al. (2020); Dong et al. (2019). In Cubuk et al. (2017), the dependence of adversarial robustness on the network architecture via NAS is investigated. A neural architecture search framework for adversarial medical image segmentation is proposed in Dong et al. (2019). Guo *et al.* Guo et al. (2020) leverages one-shot NAS Bender et al. (2018) to understand the influence of network architectures

Rungi Wang and Linin Yang have contributed equally to this work.

✉ Baochang Zhang
bczhang@buaa.edu.cn

1 Beihang University, Beijing, China
2 Zhongguancun Laboratory, Beijing, China
3 National University of Singapore, Singapore, Singapore
4 Buffalo University, Buffalo, USA
5 Nanchang Institute of Technology, Nanchang, China

against adversarial attacks. Although promising performance is achieved in the existing NAS-based methods, there is not search strategy has been proposed to improve the robustness of network architecture. In this paper, we explores NAS with adversarial optimization and more diverse operations, such as denoising blocks, weight-free operations, Gabor filters and convolutions, to improve its robustness to the adversarial attacks.

The low search efficiency has prevented NAS from its practical use, and the introduction of adversarial optimization and a larger search space further exacerbates the issue. Early works directly regard network architecture search as a black-box optimization problem in a discrete search space and take thousands of GPU days. To reduce the search space, a common idea is to adopt a cell-based search space Zoph et al. (2018). However, when it comes to searching in a huge and complicated search space, prior cell-based works may still suffer from the memory issues and are computationally intensive with the number of meta-architecture. For example, DARTS Liu et al. (2018) can only optimize over a small subset of 8 cells, which are then stacked together to form a deep network of 20. To increase search efficiency, we reformulate NAS as a multi-armed bandit problem with a comprehensive search space. The multi-armed bandit algorithm aims to determine the optimal arm selection that maximizes the reward. The reward is determined based on the selection without state transition. When an arm is selected, it produces a reward return, whereas an unselected arm does not produce any reward. Likewise, the selection or not of each potential operation is at the heart of NAS to achieve a new neural architecture. Only the selected operation contributes to the final architecture and affects the network's performance. The selected operation in NAS is comparable to the selected arm in the multi-armed bandit algorithm, as both generate a reward return without a state transition. By analogy, NAS is highly intuitive from an arm selection and a reward return point of view and hence the multi-armed bandit algorithm is a logical fit for NAS. Besides, the multi-armed bandit algorithm targets at predicting the best arm at a sequence of trials, to balance the result and its uncertainty. Likewise, NAS aims to get the best operation from an operation pool at each edge of the model with finite steps of optimization, which is also similar to the multi-armed bandit algorithm. They are both exploration and exploitation problems. Therefore, we tried to introduce the multi-armed bandit algorithm into NAS. Furthermore, the multi-armed bandit algorithm avoids the gradient descent process and provides promising search speed for NAS. Unlike traditional Upper Confidence Bound (UCB) bandit algorithms that prefer sampling using UCB and emphasizing exploration, we propose Anti-Bandit to further exploit both UCB and Lower Confidence Bound (LCB) to balance exploration and exploitation. We achieve an accuracy-bias trade-off for the operation performance estimation during the search process. It is desirable to use the trial performances to identify the optimal architecture quickly. With the help of the Anti-Bandit algorithm, our Anti-Bandit NAS (ABanditNAS) can handle the huge and complicated search space, where the number of operations that define the space can be 9^{60}!

During the operation selection of NAS, our proposed Anti-Bandit algorithm uses UCB to reduce the search space, and LCB to ensure that each arm is fairly and adequately tested before being abandoned as shown in Fig. 1. Operations being tested for their effects should not only be chosen based on their current performance (exploitation) but also on their potential for better performance (exploration). For LCB, operations with a low LCB typically suggest that they are tested insufficiently or contribute little to the architecture. These operations are chosen for exploration as we aim to confirm their contributions with sufficient tests. For UCB, the operations with the lowest UCB after sufficient trials are seldomly potential whilst the remaining operations are still exploitable. The operations with the lowest UCB can be pruned based on the following observation. The early optimal operation is not always the best one in the end but that the worst operations in the early stage typically have poorer performance at the end Zheng et al. (2019). Through pruning, we keep exploiting the operations with high reward potential, i.e., the reserved operations. To balance exploration and exploitation, we aim to encourage every arm to be fairly and adequately tested based on LCB before being abandoned based on UCB. With more testing, the operation's variance will decrease, while UCB and LCB are becoming closer to the expectation value as illustrated in Eqs. 9 and 12. Therefore, we establish the connection between LCB/UCB and exploration/exploitation, and find the balance accordingly. Specifically, operations which have poor performance early, such as parameterized operations, will be given more chances but abandoned once they are confirmed to be bad. Meanwhile, weight-free operations will be compared with parameterized operations only when they are well trained. On the other hand, with the operation pruning process, the search space becomes smaller and smaller, leading to an efficient search process.

Our framework shown in Fig. 1 highlights ABanditNAS for finding a robust architecture from a complicated search space. The main contributions of this paper can be summarized as follows:

– We reformulate NAS as a multi-bandit problem and introduce an Anti-Bandit algorithm based on a specific operation search strategy with a lower and an upper bound. With the Anti-Bandit algorithm, adversarial optimization and a comprehensive operation space, a new NAS framework (ABanditNAS) is developed to solve the

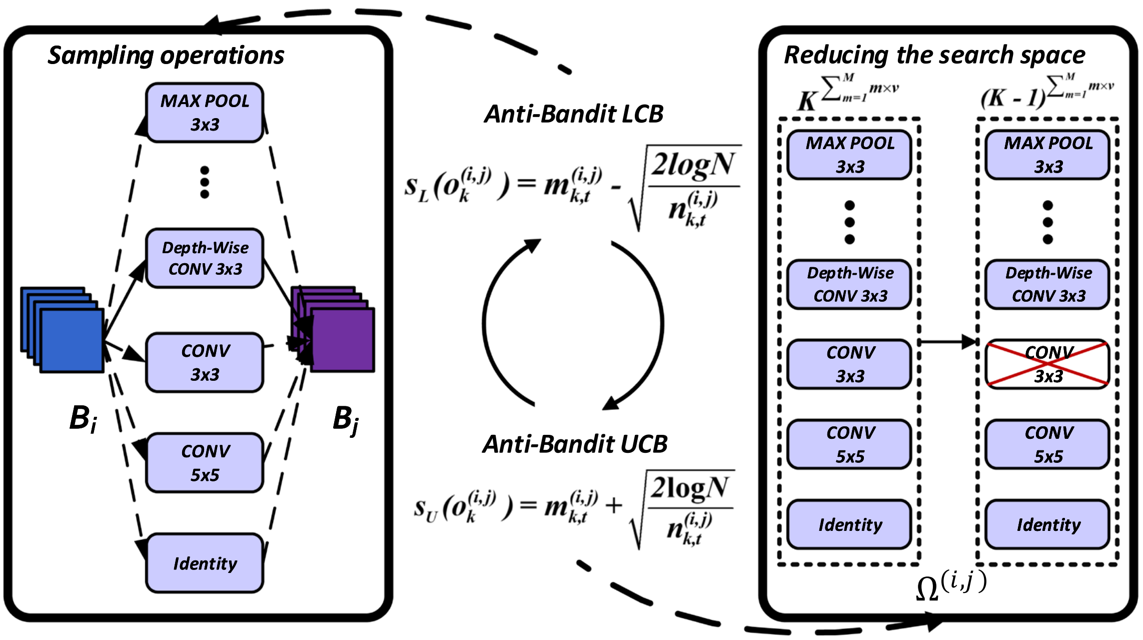

Sampling operations
MAX POOL 3x3
Depth-Wise CONV 3x3
CONV 3x3
CONV 5x5
Identity
B_i
B_j
Anti-Bandit LCB
$s_L(o_k^{(i,j)}) = m_{k,t}^{(i,j)} - \sqrt{\frac{2\log N}{n_{k,t}^{(i,j)}}}$
Anti-Bandit UCB
$s_U(o_k^{(i,j)}) = m_{k,t}^{(i,j)} + \sqrt{\frac{2\log N}{n_{k,t}^{(i,j)}}}$
Reducing the search space
$K^{\sum_{m=1}^{M} m\times v}$
$(K-1)^{\sum_{m=1}^{M} m\times v}$
$\Omega^{(i,j)}$

Fig. 1 ABanditNAS is mainly divided into two steps: sampling using LCB and abandoning using UCB

adversarially robust optimization and architecture search problems in a unified framework.

– The search space is greatly reduced by the Anti-Bandit based method by abandoning the operations with less potential, and significantly reducing the search complexity from exponential $\mathcal{O}(K^{|\mathcal{E_M}|\times v})$ to polynomial $\mathcal{O}(K^2 \times T)$, as shown in Section 3.4.
– Extensive experiments demonstrate that our proposed algorithm achieves better performance than prior NAS works, and also adversarially robust for 3 benchmarks.

Preliminary versions of this paper were published in Chen et al. (2020), which described the strategy of Anti-Bandit NAS. The new contributions of this paper include:

– The presented ABanditNAS is theoretically analyzed and the boundedness of which is proved.
– The performance of ABanditNAS in general neural architecture search is investigated in an extensive experimental section.
– A thorough analysis is conducted by comparing Anti-Bandit with the baselines Bandit and Bandit (Pruning), and the effect on the hyperparameters for NAS is shown.

2 Related Work

Bandit In probability theory, multi-armed bandit is a problem in which a decision must be made among competing choices in a way that maximizes their expected gains. A lot of optimization methods have been explored to solve bandit problems and construct optimal selection policies with the largest convergence rate Lai and Robbins (1985). For example, Q-learning and state-action-reward-state-action (SARSA) Even-Dar *et al.* (2006); Rummery and Niranjan (1994) are proposed respectively to improve classical reinforcement learning methods. Recently, AlphaGo Silver *et al.* (2017) modifies the original UCB multi-armed bandit policy by approximately predicting good arms at the start of a sequence of trials, to balance the result of simulation and its uncertainty. Inspired by other bandit methods, we propose Anti-Bandit and achieve fine adjustment by focusing on the arm with fewer trails. By fully exploiting LCB and UCB, the proposed Anti-Bandit reduces the search space and guarantees that every arm is fairly tested before being abandoned.

Neural Architecture Search (NAS) NAS becomes one of the most promising technologies in the deep learning paradigm. The early NAS works Zoph et al. (2018); Zoph and Le (2016) train and evaluate neural networks across 500 GPUs over 4 days. The following work Real et al. (2018), AmoebaNet, replaces the optimization algorithm with regularized evolution to improve the training efficiency. However, the hardware requirements and training time are still very demanding. Due to the large architecture search space, efficiency is still a key bottleneck preventing NAS from its practical use. Differentiable architecture search methods like Liu et al. (2018); Pham et al. (2018); Xue et al. (2021) formulate the task in a differentiable manner and consider each architecture in the search space as a sub-graph sampled from a super-graph. With this framework, they can accelerate the search process through parameter sharing. Besides, Sheth et al. Sheth and Xie (2023) propose a new approach for improving the generalizability and stability of differentiable NAS, by developing a transferability-encouraging tri-level optimization framework that improves the architecture of a main model by encouraging good transferability to an auxiliary model. Zhang et al. Zhang et al. (2023) introduce a neural architecture search framework for graph transformer, which automatically discovers the optimal graph transformer architectures by joint optimization of transformer architecture and graph encoding strategies. Apart from differentiable architecture search methods, another alternative to model NAS is Reinforcement Learning (RL) Zoph and Le (2016); Cai et al. (2018); Zhou *et al.* (2022). RL has an advantage for searching with complicated search space and is a goal-oriented optimization method driven by impact response or the signal. Recently, the network optimized by RL is proven to have a stronger generalization and is suitable for multiple tasks Pinto et al. (2023). The first NAS work Zoph and Le (2016) adopts RNN as the controller and policy gradient to maximize the expected reward of the controller sampling network. Cai et al. Cai et al. (2018) propose a method that regards the search space as a tree structure for network transformation. In this method, new network architectures can be generated by a father network with certain predefined transformations, which reduces the search space and speeds up the search. Besides, RL-NAS Zhou *et al.* (2022) adaptively and automatically designs high-accuracy network models according to the intended diagnosis tasks. A weight-sharing mechanism has been developed to alleviate the bottleneck of NAS where the computation time increases exponentially with the expansion of the search space and the deepening of the model structure. However, this searching strategy of RL is time-consuming. Differently, to further improve the efficiency, we reformulate NAS as a multi-armed bandit problem and present a bandit optimization algorithm for NAS. Compared with other algorithms, the proposed algorithm determines the operation based on exploration and exploitation, leading to a competitive performance at a reduced cost on complicated search space.

Model Defense Recent studies have revealed the vulnerability of neural networks to adversarial examples. Taking image classification as an example. If some small intentional pixel changes in an image which cannot be identified by the naked eye, may fool the classifier into making a false prediction. Moreover, adversarial examples often exhibit black-box transfer, meaning that adversarial examples for one model can fool another model. Therefore, it is essential for the models to resist adversarial attacks. An intuitive model defense method against antagonistic samples is adversarial training Kurakin et al. (2016); Na et al. (2017). This strategy tries to improve the robustness of neural networks by using adversarial samples during training. By contrast, works like Liao et al. (2018); Athalye et al. (2018); Gupta and Rahtu (2019) prefer to remove adversarial perturbation before feeding them to the network. For example, Dziugaite et al. (2016); Das et al. (2017) adopt JPEG compression and Osadchy *et al.* (2017) apply low-pass filters. As DNN is intuitively robust to random disturbances, many defenses prefer to use randomization to mitigate the input/characteristic domain's effects against the disturbance. Although randomization-based defense performs well in black-box attack settings, it fails to handle white-box attacks like EoT, which considers random processes during the attack. To improve the robustness of white-box adversarial attacks, recent works start to design new operators or architectures Zhou et al. (2019). Instead of manual design, finding adversarially robust architectures using NAS is a promising direction Cubuk et al. (2017), which is worthy of in-depth exploration. Though Cubuk *et al.* Cubuk et al. (2017) searches adversarially robust architectures via a reinforcement learning strategy and achieves favorable performance, it is still time-consuming and lacks diverse operations that are directly related to model defense.

In this paper, an Anti-Bandit algorithm is introduced into NAS, and a new optimization framework is developed to generate adversarially robust networks accordingly. Unlike Ilyas et al. (2018) using bandits to produce black-box adversarial samples, we solve the adversarially robust optimization and architecture search problems for a unified framework to obtain a robust network architecture. Compared with the existing NAS-based model defense methods Cubuk et al. (2017); Kotyan et al. (2020); Guo et al. (2020); Dong et al. (2019), the presented method in this paper not only balances robustness and search efficiency, but also improves the classification accuracy.

3 Anti-Bandit

Our goal is to search the network architectures effectively and efficiently. However, there exists a dilemma for NAS whether to keep a network structure that offers significant rewards (exploitation) or further investigate other network structures (exploration). Based on the probability theory, the aforementioned exploration-versus-exploitation dilemma can be solved by the multi-armed bandit which makes decisions among competing choices in a way that maximizes their expected gain. Specifically, we propose Anti-Bandit, which chooses and discards at trial the arm k based on

$$\tilde{r}_k - \tilde{\delta}_k \leq r_k \leq \tilde{r}_k + \tilde{\delta}_k, \tag{1}$$

where r_k, $\tilde{r}_k$ and $\tilde{\delta}_k$ are the actual reward, the average estimated reward and the estimated variance obtained from arm k. $\tilde{r}_k$ is the value term that favors actions that perform well historically, and $\tilde{\delta}_k$ is the exploration term that makes actions that get an exploration bonus. $\tilde{r}_k - \tilde{\delta}_k$ and $\tilde{r}_k + \tilde{\delta}_k$ can be interpreted as the lower bound and upper bound of a confidence interval,

Traditional UCB algorithm, which optimistically substituting $\tilde{r}_k + \tilde{\delta}$ for r_k, emphasizes the exploration, however, disregards the exploitation. Unlike UCB bandit, we further exploit both LCB and UCB to balance exploration and exploitation. A smaller LCB usually has small expectations but a large variance and should be given a bigger chance to be sampled for more trials. Following, based on the observation that the worst operations in the early stage usually have a worse performance at the end Zheng et al. (2019), we use UCB to prune the operation with the worst performance and reduce the search space. In summary, we adopt LCB, $\tilde{r}_k - \tilde{\delta}$, to sample the arm which should be further optimized and use UCB, $\tilde{r}_k + \tilde{\delta}$, to abandon the operation with the minimum value. Because the variance is bounded and converges, the operating estimate value is always near the actual value and gradually approaches the actual value as the number of trials increases. Our Anti-Bandit algorithm overcomes the limitations of an exploration-based strategy, including levels of understanding and suboptimal gaps. The definitions of the reward term $\tilde{r}_k$, variance term $\tilde{\delta}$, and the proof of our proposed method are shown in the following.

Definition 1 If an operation on arm k has been recommended n_k times, $reward_i$ is the reward on arm k in the i-th trail. The value term of Anti-Bandit is defined as follows:

$$\tilde{r}_k = \frac{\sum_{i=1}^{n_k} reward_i}{n_k}. \tag{2}$$

The value of the k-th operation $\tilde{r}_k$ is defined based on the expected reward $\sum reward_i$. If n_k approaches infinity, the $\tilde{r}_k$ approaches the actual value of operation r_k. However, the number of operations n_k can't be infinite. Therefore, we should approximate the actual value as close as possible through the variance.

Definition 2 There exists a difference between the estimated reward $\tilde{r}_k$ and the actual reward r_k, and we can estimate the variance of Anti-Bandit with

$$\tilde{\delta}_k = \sqrt{\frac{2\ln N}{n_k}}, \tag{3}$$

where N is the total number of the trail.

Proof Suppose p represents the theoretical reward of each operation which is independently distributed and $p \in [0, 1]$. n is the number of times of the arm has been played up to trial, X_i is the estimated value of the operation in i-th trail and $X = \frac{\sum_i X_i}{n}$. δ is the difference between the theoretical reward and estimated reward. Since the global variance boundary can be represented by the variance boundary of independent trials (See Appendix), based on Markov's inequality, we can arrive at below:

$$\begin{aligned} P[p > X + \delta] &= P[\sum_i (p - X_i) > n\delta] \\ &= P[e^{\lambda \sum_i (p-X_i)} > e^{n\lambda\delta}] \\ &\leq \frac{E[e^{\lambda \sum_i (p-X_i)}]}{e^{n\lambda\delta}}. \end{aligned} \tag{4}$$

Based on Hoeffding's Lemma, for any real-valued variable Y with $a \leq Y \leq b$, we have $E(e^{\lambda(E(Y)-Y)}) \leq e^{\frac{\lambda^2(b-a)^2}{8}}$. In our case, $a = 0$ and $b = 1$, we get

$$\begin{aligned} \frac{E[e^{\lambda \sum_i (p-X_i)}]}{e^{n\lambda\delta}} &= \frac{\prod_i E[e^{\lambda(p-X_i)}]}{e^{n\lambda\delta}} \leq \frac{\prod_i e^{\frac{\lambda^2}{8}}}{e^{n\lambda\delta}} \\ &= e^{\frac{n\lambda^2}{8} - n\lambda\delta}. \end{aligned} \tag{5}$$

Since λ is a positive constant, taking the derivative of $-n\lambda\delta + \frac{n\lambda^2}{8}$ as zero shows that the minimum is achieved at $\lambda = 4\delta$, *i.e.* , $\min_{\lambda > 0} e^{-n\lambda\delta + \frac{n\lambda^2}{8}} = e^{-2n\delta^2}$. Therefore, we have $P[p > X + \delta] \leq e^{-2n\delta^2}$. According to the symmetry of the distribution, we have $P[p < X - \delta] \leq e^{-2n\delta^2}$. Finally, we get the inequality:

$$P[|p - X| \leq \delta] \geq 1 - 2e^{-2n\delta^2}. \tag{6}$$

We need to make the δ decrease as the number of operation recommendations increases, therefore we choose $\sqrt{\frac{2\ln N}{n}}$ as δ. This is to say, $X - \sqrt{\frac{2\ln N}{n}} \leq p \leq X + \sqrt{\frac{2\ln N}{n}}$ is implemented at least with probability $1 - \frac{2}{N^4}$. As the trail progresses, the value of variance will gradually decrease, that X will gradually approach p. Equation 7 shows that we can achieve 0.992 probability when the number of the trail gets 4.

$$1 - \frac{2}{N^4} = \begin{array}{l} 0.875 \text{ N=2} \\ 0.975 \text{ N=3} \\ 0.992 \text{ N=4}. \end{array} \tag{7}$$

□

According to proof, the variance in the Anti-Bandit algorithm is bounded, and the lower/upper confidence bounds can be estimated as

$$\tilde{r}_k - \sqrt{\frac{2\ln N}{n}} \leq r_k \leq \tilde{r}_k + \sqrt{\frac{2\ln N}{n}}. \tag{8}$$

4 Anti-Bandit Neural Architecture Search

Compared to conventional NAS, NAS with adversarial training is more challenging due to the additional burden. For example, adversarial training using the F-step PGD attack needs roughly $F+1$ times more computation. In this section, we introduce search spaces for NAS with and without adversarial training respectively in Sec. 4.1, our Anti-Bandit strategy for NAS in Sec. 4.2 and adversarial optimization for ABanditNAS in Sec. 4.3.

4.1 Search Space

Following Zoph et al. (2018); Liu et al. (2018); Zheng et al. (2019), we search for computation cells as the building blocks of the final architecture. A cell is a fully-connected directed acyclic graph (DAG) of M nodes, *i.e.* , $\{B_1, B_2, ..., B_M\}$ as shown in Fig. 2a. Here, each node is a specific tensor (*e.g.* , a feature map in convolutional neural networks) and each directed edge (i, j) between B_i and B_j denotes an operation $o^{(i,j)}(.)$, which is sampled from $\Omega^{(i,j)} = \{o_1^{(i,j)}, ..., o_K^{(i,j)}\}$. $\{\Omega^{(i,j)}\}$ is the search space of a cell. Each node B_j takes its dependent nodes as input, and can be obtained by $B_j = \Sigma_{i<j} o^{(i,j)}(B_i)$. The constraint $i < j$ here is to avoid cycles in a cell. Each cell takes the output of the previous cell as input. For brevity, we denote B_0 as the last node of the previous cell and the first node of the current cell. Unlike existing approaches only using normal and reduction cells, we search for v ($v > 2$) cells instead. For general NAS search, we follow Liu et al. (2018) and take seven normal operations, *i.e.* , 3×3 max pooling, 3×3 average pooling, skip connection (identity), 3×3 dilated convolution with rate 2, 5×5 dilated convolution with rate 2, 3×3 depth-wise separable convolution, and 5×5 depth-wise separable convolution. Taking account of adversarially robust optimization for NAS, we introduce two additional operations, 3×3 Gabor filter and denoising block, for model defense. Therefore, the size of the whole search space is $K^{|\mathcal{E}_\mathcal{M}| \times v}$, where $\mathcal{E}_\mathcal{M}$ is the set of possible edges with M intermediate nodes in the fully-connected DAG. In the case with $M = 4$ and $v = 6$, together with the input node, the total number of cell structures in the search space is $9^{(1+2+3+4)\times 6} = 9^{10\times 6}$. Here, we briefly introduce the two additional operations.

Gabor Filter Gabor filters Gabor (1946a, b) containing frequency and orientation representations can characterize the spatial frequency structure in images while preserving spatial relationships. Gabor filters are motivated by the accuracy and robustness of the animal visual system. Based on the theoretical analysis of network robustness Cisse et al. (2017), the work Pérez et al. (2020) derives a Lipschitz constant for Gabor filters and shows that networks with Gabor filters yield good robustness for adversarial attacks Pérez et al. (2020); Dapello et al. (2020). We add Gabor filters to our search space to find a robust architecture. Gabor filters are defined as: $\exp(-\frac{x'^2+\gamma^2 y'^2}{2\sigma^2}) \cos(2\pi \frac{x'}{\lambda} + \psi)$. Here, $x' = x\cos\theta + y\sin\theta$ and $y' = -x\sin\theta + y\cos\theta$. σ, γ, λ, ψ and θ are learnable parameters. Note that the symbols used here apply only to the Gabor filter and are different from the symbols used in the rest of this paper. Figure 2b shows an example of Gabor filters.

Denoising Block As described in Xie et al. (2019), adversarial perturbations on images will introduce noise in the features. Therefore, denoising blocks can be used to improve adversarial robustness via feature denoising. Following this, we add the non-local mean denoising block Buades et al. (2005) as shown in Fig. 2c to the search space to denoise the features. It computes a denoised feature map z of an input feature map x by taking a weighted mean of the features' overall spatial locations $\mathcal{L}$ as $z_p = \frac{1}{C(x)} \sum_{\forall q \in \mathcal{L}} f(x_p, x_q) \cdot x_q$, where $f(x_p, x_q)$ is a feature-dependent weighting function and $C(x)$ is a normalization function.

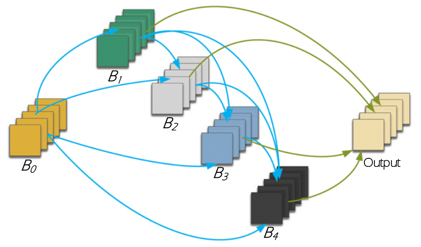
4.2 Anti-Bandit Strategy for NAS

As described in Ying et al. (2019); Zheng et al. (2019), the validation accuracy ranking of different network architectures is not a reliable indicator of the final architecture quality. However, the experimental results actually suggest a nice property that if an architecture performs poorly in the beginning of training, there is little hope that it can be part of the final optimal model Zheng et al. (2019). As the training progresses, this observation is more and more certain. Based on this observation, we derive a simple yet effective training strategy. During training, along with the increasing epochs, we progressively abandon the worst performing operation and sample the operations with small expectations but the large variance for each edge. Unlike Zheng et al. (2019) which just uses the performance as the evaluation metric to decide which operation should be pruned, we use the Anti-Bandit algorithm described in Sec. 3 to make a decision.

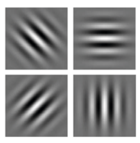
Following UCB in the bandit algorithm, we obtain the initial performance for each operation on every edge. Specifically, we sample one from the K operations in $\Omega^{(i,j)}$ for every edge, then obtain the validation accuracy a which is the initial performance $m_{k,0}^{(i,j)}$ by training adversarially the sampled network for one epoch, and finally assigning this accuracy to all the sampled operations.

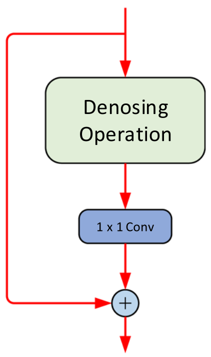
By considering the confidence of the k-th operation using Eq. 8, the LCB is calculated by

$$s_L(o_k^{(i,j)}) = m_{k,t}^{(i,j)} - \sqrt{\frac{2\log N}{n_{k,t}^{(i,j)}}}, \tag{9}$$



(a)


(b)



(c)

Fig. 2 **a** A cell containing four intermediate nodes B_1, B_2, B_3, B_4 that apply sampled operations on the input node B_0. B_0 is from the output of the previous cell. The output node concatenates the outputs of the four intermediate nodes. **b** Gabor Filter. **c** A generic denoising block. Following Xie et al. (2019), it wraps the denoising operation with a 1×1 convolution and an identity skip connection He et al. (2016)

Algorithm 1: ABanditNAS with adversarial training

```
Input: Training data, validation data, searching hyper-graph,
       adversarial perturbation $\delta$, adversarial manipulation
       budget $\epsilon$, $K = 9$, hyper-parameters $\alpha$, $\lambda = 0.7$, $T = 3$.
Output: The remaining optimal structure;
1  $t = 0$; $c = 0$;
2  Get initial performance $m_{k,0}^{(i,j)}$;
3  while ($K > 1$) do
4      $c \leftarrow c + 1$;
5      $t \leftarrow t + 1$;
6      Calculate $s_L(o_k^{(i,j)})$ using Eq. 9;
7      Calculate $p(o_k^{(i,j)})$ using Eq. 10;
8      Select an architecture by sampling one operation based on
       $p(o_k^{(i,j)})$ from $\Omega^{(i,j)}$ for every edge;
9      # Train the selected architecture adversarially:
10     for $e = 1, ..., E$ do
11         $\delta = \text{Uniform}(-\epsilon, \epsilon)$;
12         $\delta \leftarrow \delta + \alpha \cdot \text{sign}\left(\nabla_x l\big(f(x_e + \delta), y_e\big)\right)$;
13         $\delta = \max\left(\min(\delta, \epsilon), -\epsilon\right)$;
14         $\theta \leftarrow \theta - \nabla_\theta l\big(f_\theta(x_e + \delta), y_e\big)$.
15     end
16     Get the accuracy $a$ on the validation data;
17     Update the performance $m_{k,t}^{(i,j)}$ using Eq. 11;
18     if $c = K * T$ then
19         Calculate $s_U(o_k^{(i,j)})$ using Eq. 12;
20         Update the search space $\{\Omega^{(i,j)}\}$ using Eq. 13;
21         $c = 0$;
22         $K \leftarrow K - 1$.
23     end
24 end
```

where N is to the total number of samples, $n_{k,t}^{(i,j)}$ refers to the number of times the kth operation of edge (i, j) has been selected, and t is the index of the epoch. The first item in Eq. 9 is the value term (See Eq. 2) which favors the operations that look good historically and the second is the exploration term (See Eq. 3) which allows operations to get an exploration bonus that grows with $\log N$. The selection probability for each operation is defined as

$$p(o_k^{(i,j)}) = \frac{\exp\{-s_L(o_k^{(i,j)})\}}{\sum_m \exp\{-s_L(o_m^{(i,j)})\}}. \tag{10}$$

The minus sign in Eq. 10 means that we prefer to sample operations with a smaller confidence. After sampling one operation for every edge based on $p(o_k^{(i,j)})$, we obtain the validation accuracy a by training adversarially the sampled network for one epoch, and then update the performance $m_{k,t}^{(i,j)}$ which historically indicates the validation accuracy of all the sampled operations $o_k^{(i,j)}$ as

$$m_{k,t}^{(i,j)} = (1 - \lambda) m_{k,t-1}^{(i,j)} + \lambda * a, \tag{11}$$

where λ is a hyper-parameter.

Finally, after $K * T$ samples where T is a hyper-parameter, we calculate the confidence with the UCB according to Eq. 8 as

$$s_U(o_k^{(i,j)}) = m_{k,t}^{(i,j)} + \sqrt{\frac{2 \log N}{n_{k,t}^{(i,j)}}}. \tag{12}$$

The operation with the minimal UCB for every edge is abandoned. This means that the operations that are given more opportunities but result in poor performance are removed. With this pruning strategy, the search space is significantly reduced from $|\Omega^{(i,j)}|^{10 \times 6}$ to $(|\Omega^{(i,j)}| - 1)^{10 \times 6}$, and the reduced space becomes

$$\Omega^{(i,j)} \leftarrow \Omega^{(i,j)} - \{\mathop{\arg\min}_{o_k^{(i,j)}} s_U(o_k^{(i,j)})\}, \forall (i, j). \tag{13}$$

The reduction procedure is carried out repeatedly until the optimal structure is obtained where there is only one operation left on each edge.

Complexity Analysis There are $\mathcal{O}(K^{|\mathcal{E}_\mathcal{M}| \times v})$ combinations in the process of finding the optimal architecture in the search space with v kinds of different cells. In contrast, ABanditNAS reduces the search space for every $K * T$ epoch. Therefore, the complexity of the proposed method is

$$\mathcal{O}(T \times \sum_{k=2}^{K} k) = \mathcal{O}(TK^2). \tag{14}$$

4.3 Adversarial Optimization

The goal of adversarial training Madry et al. (2017) is to learn networks with robustness to adversarial attacks. Given a network f_θ parameterized by θ, a dataset (x_e, y_e), a loss function l and a threat model Δ, the learning problem can be formulated as following optimization problem: $\min_\theta \sum_e \max_{\delta \in \Delta} l\big(f_\theta(x_e + \delta), y_e\big)$, where δ is the adversarial perturbation. In this paper, we consider the typical l_∞ threat model Madry et al. (2017), $\Delta = \{\delta : \|\delta\|_\infty \leq \epsilon\}$ for some $\epsilon > 0$. Here, $\|\cdot\|_\infty$ is the l_∞-norm distance metric and ϵ is the adversarial manipulation budget. The procedure for adversarial training is to use attacks to approximate the inner maximization over Δ, followed by some variation of gradient descent on the model parameters θ. For example, one of the earliest versions of adversarial training uses the Fast Gradient Sign Method (FGSM) Goodfellow et al. (2015) to approximate the inner maximization. This could be seen as a relatively inaccurate approximation of the inner maximization for l_∞ perturbations, and it has the closed form solution: $\theta = \epsilon \cdot \text{sign}\Big(\nabla_x l\big(f(x), y\big)\Big)$. A better approximation of the inner maximization is to take multiple, smaller FGSM steps of size α instead. However, the number of gradient computations caused by the multiple steps is proportional to $\mathcal{O}(EF)$ in a single epoch, where E is the size of the dataset and F is the number of steps taken by the PGD adversary. This is F times greater than the standard training which has $\mathcal{O}(E)$ gradient computations per epoch, and the adversarial training is typically F times slower. To accelerate the adversarial training, we combine the FGSM with random initialization Wong et al. (2020) for our ABanditNAS. Our ABanditNAS with adversarial training is summarized in Algorithm 1.

5 Experiments

Our anti-bandit strategy searches for net architecture through probabilistic sampling, which is an optimization algorithm in search process. In this section, we verify the effectiveness and efficiency of ABanditNAS in the general NAS setting and the adversarial setting with three benchmark datasets (MNIST LeCun *et al.* (1998), CIFAR-10 Krizhevsky et al. (2009) and ImageNet-1k Deng et al. (2009)) for the image classification task, and compare ABanditNAS with prior methods. The bold values in the Tables 1–5 represent the best performance.

5.1 Experiment Protocol

In our experiments, we search architectures with an over-parameterized network, and then evaluate the best architecture on corresponding datasets. We run each experiment multiple times and find that the resulting architectures only show a slight variation in performance, which demonstrates the stability of our method. For fair comparison, all GPU days are based on the Titan V GPU.

Baseline We also apply the conventional bandit which samples operations based on UCB to search the network, leading to UCBNAS. The main differences between UCBNAS and ABanditNAS lie in that UCBNAS only uses UCB as an evaluation measure to select an operation, and there is no operation pruning involved. Also, to further demonstrate the effectiveness of our ABanditNAS, we use UCBNAS (pruning) to search for a robust model. Compared to our ABanditNAS and UCBNAS, UCBNAS (pruning) takes the pruning process based on the UCB and ignores the usage of LCB for pruning.

White-Box versus Black-Box Attack Settings In an adversarial setting, there are two main threat models: white-box attacks where the adversary possesses complete knowledge of the target model, including its parameters, architecture and the training method, and black-box attacks where the adversary feeds perturbed images at test time, which are generated without any knowledge of the target model, and observes the output. We evaluate the robustness of our proposed defense against both settings. The perturbation size ϵ and step size are the same as those in the adversarial training for both the white-box and black-box attacks. The numbers of iterations for MI-FGSM, BIM and PGD are set to 10, 10 and 7 with step sizes and a standard perturbation size the same as those in the white-box attacks. We evaluate ABanditNAS against transfer-based attacks where a copy of the victim network is trained with the same training setting. We apply attacks similar to the white-box attacks on the copied network to generate black-box adversarial examples. We also generate adversarial samples using a ResNet-18 model and feed them to the model obtained by ABanditNAS.

5.2 Settings and Results of General NAS

Settings We did experiments on clean data sets without confrontation about ABanditNAS and UCBNAS (pruning) on

Table 1 Comparison with state-of-the-art architectures on CIFAR10. ‘channel=72’ means we train a larger network with 72 initial channels

Architecture	Accuracy (%)	# Params (M)	Search cost (GPU days)	Search Method
DenseNet-BC Huang et al. (2017)	96.54	25.6	–	Manual
NASNet-A Zoph et al. (2018)	97.35	3.3	1800	RL
AmoebaNet-A Real et al. (2018)	97.45 ± 0.06	3.2	3150	Evolution
AmoebaNet-B Real et al. (2018)	97.45 ± 0.05	2.8	3150	Evolution
PNAS Liu et al. (2018)	96.59 ± 0.09	3.2	225	SMBO
ENAS Pham et al. (2018)	97.11	4.6	0.5	RL
DARTS (1st order) Liu et al. (2018)	97.00 ± 0.14	3.3	0.4	Gradient
DARTS (2nd order) Liu et al. (2018)	97.24 ± 0.09	3.3	1	Gradient
SNAS (mild) Xie et al. (2018)	97.02	2.9	1.5	Gradient
ProxylessNAS Cai et al. (2019)	97.92	3.3	4	Gradient
P-DARTS Chen et al. (2019)	97.5	3.3	0.3	Gradient
PC-DARTS Xu et al. (2019)	97.43 ± 0.07	3.3	0.1	Gradient
SGAS Li et al. (2020)	97.34 ± 0.24	3.7	0.5	Gradient
SDARTS-RS Chen and Hsieh (2020)	97.33	3.4	0.4	Gradient
SDARTS-ADV Chen and Hsieh (2020)	97.39	3.3	1.3	Gradient
UCBNAS (pruning)	96.89	–	**0.09**	Bandit
ABanditNAS	97.13	**3.0**	**0.09**	Anti-Bandit
ABanditNAS (channel=72)	**97.58**	11.6	**0.09**	Anti-Bandit

CIFAR-10 and ImageNet-1k. Following Xu et al. (2019), we search a large network with 6 cells, which contains 4 normal cells and 2 reduction cells. The reduction cells are located at 2-th and 4-th of the total depth of the network, and the rest are normal cells. We take the seven normal operations for NAS search as shown in Sec. 4.1. There are 6 nodes with 4 intermediate nodes, 1 input note and 1 output node. The output node of a cell is the depth-wise concatenation of all the intermediate nodes. Thus, 14 edges in each cell during the search, and the search space of a cell consists of operations on all edges. We set the momentum to 0.9, the weight decay to 3×10^4 and the initial learning rate to 0.1. The hyper-parameter T which denotes the sampling times is set to 4, and the batch size is 384. The initial number of channels is 16. We use 90% of the training set as the training data and 10% of the training set as the validation data in the search phase. After the search, we train the network with a batch size of 96.

The other hyper-parameters remain the same as the search phase. We train the finial architecture for 600 epochs on CIFAR-10. For ImageNet-1k, as it is much more difficult than CIFAR10. We modify the network architecture used on CIFAR-10 to fit ImageNet-1k with 48 initial channels. The network is trained for 250 epochs with a batch size of 512. The SGD optimizer with an initial learning rate of 0.25, a momentum of 0.9 and a weight decay of 3×10^5 is used.

CIFAR-10 Table 1 shows the comparison with state-of-the-art architectures on CIFAR10. Fig. 3 is detailed structures of the best cells discovered. We use one Titan V GPU to search, and the batch size is 384. The entire search process takes about 0.09 GPU days (2.25 hours). In addition, we also train a larger network with 72 initial channels on the same architecture as shown in Fig. 3. It can clearly be seen in Table 1. Compared with DARTS (1st order), ABanditNAS achieves not only a better performance (97.00% vs. 97.13%), but also fewer parameters (3.3M vs. 3.0M). When compared with UCBNAS (pruning), ABanditNAS achieves better performance (96.89% vs. 97.13%) in the same search time.

Compared with all NAS methods in Table 1, ABanditNAS search speed is the fastest. Although, ABanditNAS underperforms several gradient-based methods (such as ProxylessNAS), we intuitively analysis that ABanditNAS uses the reward of each operation as the sampling probability and adds pruning process, which might be worse for a small search space but benefit for a bigger one. CIFAR-10 is somewhat simple, and the performance gap between each operation is not obvious. Therefore, we further verify the performance of ABanditNAS on the ImageNet-1k.

ImageNet-1k Table 2 shows the comparison with state-of-the-art architectures on ImageNet-1k. UCBNAS (pruning) is the network searched and pruned by UCB method. Moreover, PC-DARTS (ImageNet-1k) and ABanditNAS (ImageNet-1k) represent the network searched on ImageNet-1k dataset, but otherwise the networks are searched on CIFAR10. The result of ABanditNAS searched on CIFAR10 outperforms other prior results, even the result of PC-DARTS searched on ImageNet-1k (75.76% vs. 75.62%), and the result of ABanditNAS searched on ImageNet-1k further improves to

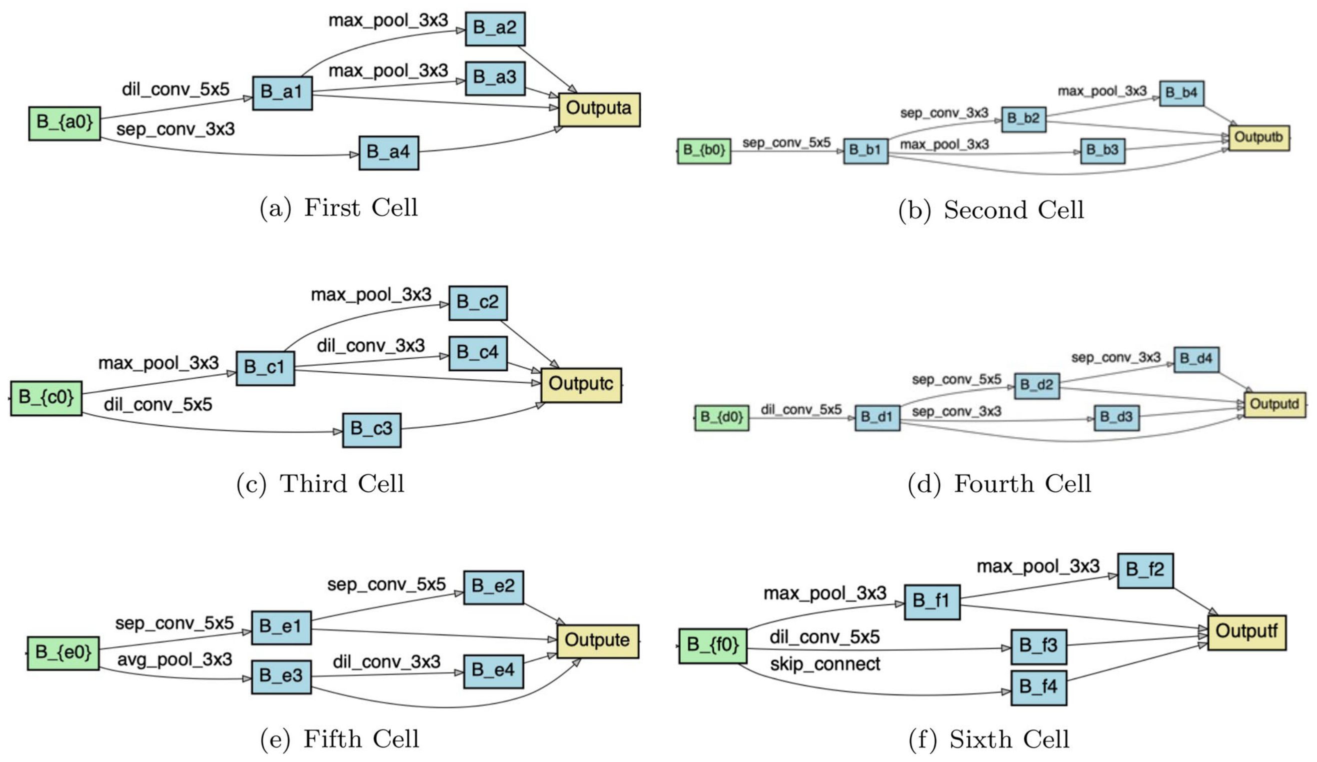


Fig. 3 Detailed structures of the best cells discovered on CIFAR-10

Table 2 Comparison with state-of-the-art architectures on ImageNet-1k

Architecture	Accuracy(%) top-1	top-5	# Params (M)	Search cost (GPU days)	Search Method
Inception-v1 Szegedy et al. (2015)	69.8	89.9	6.6	–	Manual
MobileNet Howard et al. (2017)	70.6	89.5	4.2	–	Manual
ShuffleNet 2x (v1) Zhang et al. (2018)	75.6	89.8	~ 5	–	Manual
ShuffleNet 2x (v2) Ma et al. (2018)	74.9	–	~ 5	–	Manual
NASNet-A Zoph et al. (2018)	74.0	91.6	5.3	1800	RL
AmoebaNet-C Real et al. (2018)	75.7	92.4	6.4	3150	Evolution
PNAS Liu et al. (2018)	74.2	91.2	5.1	225	SMBO
DARTS (2nd order) Liu et al. (2018)	73.2	91.3	4.7	4.0	Gradient
SNAS (mild) Xie et al. (2018)	72.7	90.8	4.3	1.5	Gradient
ProxylessNAS Cai et al. (2019)	75.1	92.5	7.1	8.3	Gradient
P-DARTS Chen et al. (2019)	75.6	92.6	4.9	0.3	Gradient
PC-DARTS Xu et al. (2019)	74.9	92.2	5.3	0.1	Gradient
PC-DARTS (ImageNet-1k) Xu et al. (2019)	75.8	92.7	5.3	3.8	Gradient
SGAS Li et al. (2020)	75.62	92.61	5.4	0.25	Gradient
SDARTS-RS Chen and Hsieh (2020)	74.4	91.8	5.7	0.4	Gradient
SDARTS-ADV Chen and Hsieh (2020)	74.8	92.2	5.3	1.3	Gradient
UCBNAS (pruning)	69.4	90.8	**4.0**	**0.08**	Bandit
ABanditNAS	75.76	92.47	5.9	**0.08**	Anti-Bandit
ABanditNAS (ImageNet-1k)	**76.11**	**92.94**	6.1	1.5	Anti-Bandit

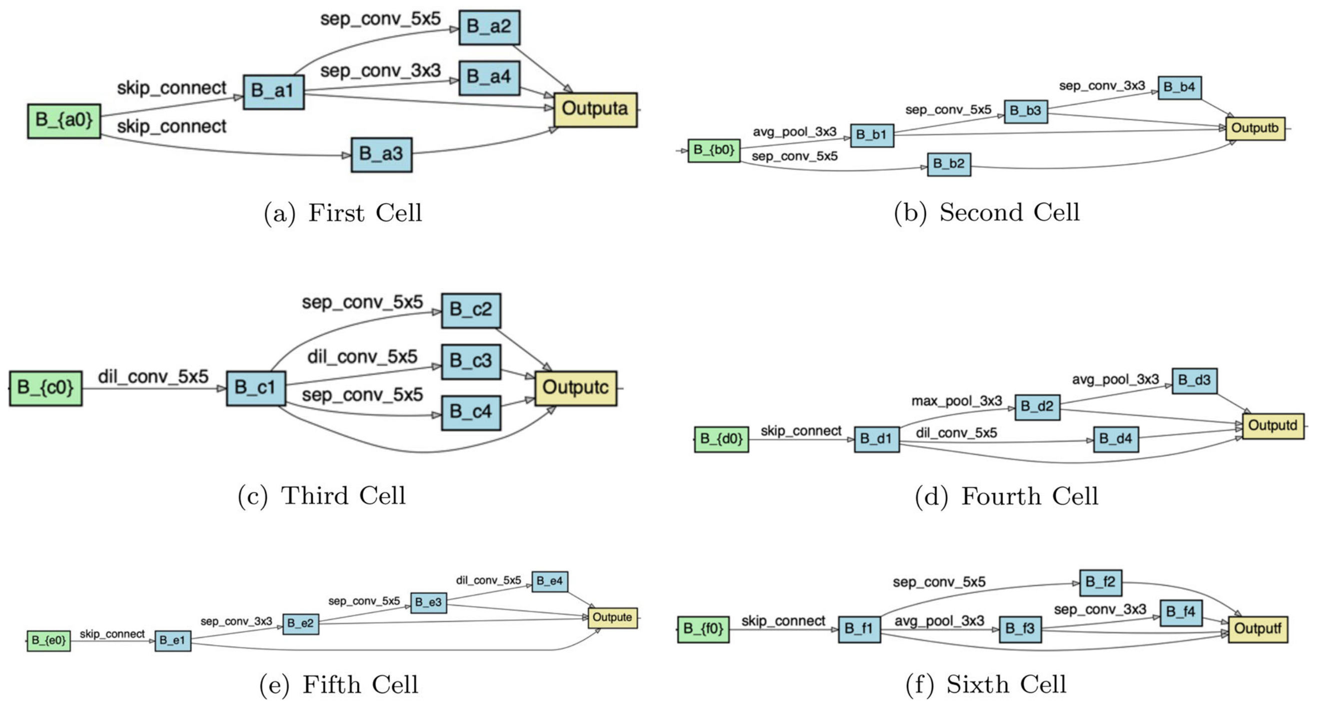


Fig. 4 Detailed structures of the best cells discovered on ImageNet-1k

76.11%. Fig. 4 is detailed structures of the best cells discovered.

5.3 Settings and Results of NAS with Adversarial Training

Settings In the search process, the over-parameterized network is constructed with six cells, where the 2-nd and 4-th cells are used to double the channels of the feature maps and halve the height and width of the feature maps, respectively. There are $M = 4$ intermediate nodes in each cell. The hyperparameter T which denotes the sampling times is set to 3, so the total number of epochs is $\sum_{k=2}^{K} k * T$. The hyperparameter λ is set to 0.7. For the search space, we use the seven normal operations, 3×3 Gabor filter and denoising block. Meanwhile, we use an additional regularization cutout DeVries and Taylor (2017) for CIFAR-10. After the search, the six cells are stacked to get the final networks. In the adversarially training process, we employ FGSM combined with random initialization and $\epsilon = 0.3$ on MNIST, and use PGD-7 with $\epsilon = 0.031$ and step size of 0.0078 on CIFAR-10. Next, we use ABanditNAS-V to represent ABanditNAS with V cells in the training process. The number V can be different from the number v. For MNIST, the initial number of channels is 16, the batch size is 512 and SGD optimizer with an initial learning rate of 0.025 is adopted. For CIFAR-10, the initial number of channels is 48, the batch size is 96 and SGD optimizer with an initial learning rate of 0.1 is applied. For both datasets, we train 150 epochs with a momentum of 0.9, a weight decay of 3×10^{-4}, and a gradient clipping at 5 and the learning rate anneals down to zero following a cosine schedule without restart. This section compares ABanditNAS to adversarial training methods (like TRADES) and NAS-based methods (like PC-DARTS).

textitMNIST Owing to the search space reduction by Anti-Bandit, the entire search process only requires 1.93 hours on a single NVIDIA Titan V GPU. For MNIST, the structure searched by ABanditNAS is directly used for training. We evaluate the trained network by 40 and 100 attack steps, and compare our method with LeNet Madry et al. (2017), LeNet (Prep. + Adv. train) Yang et al. (2019), TRADES DARTS and PCDARTS in Table 3. The backbone of TRADES is ResNet12. The search cost of UCBNAS (pruning) is less than UCBNAS, which indicates the efficiency of the pruning by UCB. From these results, we can see that ABanditNAS using FGSM adversarial training with random initialization is more robust than LeNet with PGD-40 adversarial training, no matter which attack is used. Even using matrix estimation (LeNet (Prep. + Adv. train)) as preprocessing to destroy the adversarial structure of the noise for LeNet based on Yang et al. (2019), our method still performs better. In addition, our method has the best performance (99.52%) on the clean images with a strong robustness. For the black-box attacks, Table 4 shows that they barely affect the structures searched by ABanditNAS compared with other models, either manually designed or searched by NAS. In Fig. 5a, we report more

Table 3 Robustness of ABanditNAS under FGSM and PGD attacks on MNIST.

Architecture	Clean (%)	FGSM (%)	PGD-40 (%)	PGD-100 (%)	# Params (M)	Search Cost (GPU days)	Search Method
LeNet Madry et al. (2017)	98.8	95.6	93.2	91.8	3.27	–	Manual
LeNet (Prep. + Adv. train Yang et al. (2019))	97.4	–	94.0	91.8	3.27	–	Manual
TRADES Zhang et al. (2019)	99.25	96.67	93.38	92.83	12.0	–	Manual
DARTS Liu et al. (2018)	99.21	98.12	94.89	94.77	0.274	0.56	Gradient
PC-DARTS Xu et al. (2019)	99.38	98.72	96.83	95.02	0.269	0.16	Gradient
UCBNAS	99.5	98.67	96.94	95.4	0.082	0.13	Bandit
UCBNAS (pruning)	99.52	98.56	96.62	94.96	0.066	0.08	Bandit
ABanditNAS-6	**99.52**	**98.94**	**97.01**	**95.70**	**0.089**	**0.08**	Anti-Bandit

'(Prep. + Adv. train)' indicates to preprocess and adversarial train the network based on Yang et al. (2019)

Table 4 Robustness of our model in the white-box and black-box settings on MNIST and CIFAR-10.

		White-Box Setting				Black-Box Setting			
Structure	Clean	FGSM	MI-FGSM	BIM	PGD	FGSM	MI-FGSM	BIM	PGD
	MNIST ($\epsilon = 0.3$)								
LeNet Madry et al. (2017) (copy)	98.8	95.6	–	–	93.2	96.8	–	–	96.0
ABanditNAS-6 (copy)	**99.52**	**98.94**	97.41	97.63	**97.58**	**98.86**	99.09	99.12	**99.02**
	CIFAR-10 ($\epsilon = 0.031$)								
Wide-ResNet Madry et al. (2017) (copy)	87.3	56.1	–	–	50.0	67.0	–	–	64.2
TRADES Zhang et al. (2019) (copy)	85.72	53.82	–	–	47.98	68.52	–	–	68.73
NASNet Cubuk et al. (2017) (copy)	**93.2**	63.6	–	–	50.1	78.1	–	–	75.0
DARTS Liu et al. (2018) (copy)	86.11	55.21	–	–	49.72	63.98	–	–	60.11
PC-DARTS Xu et al. (2019) (copy)	86.43	54.98	–	–	49.96	63.77	–	–	60.17
ABanditNAS-6 (copy)	87.16	55.88	48.77	47.59	50.04	76.12	74.94	75.78	76.13
ABanditNAS-6 (ResNet-18)	87.16	55.88	48.77	47.59	50.04	77.41	77.06	77.63	78.03
ABanditNAS-10 (ResNet-18)	90.64	**81.38**	**54.19**	**55.31**	**58.74**	**80.34**	**80.25**	**80.80**	**81.26**

Here ϵ is the perturbation size. PGD means PGD-40 for MNIST and PGD-7 for CIFAR-10. 'copy' means we use a copied network to generate black-box adversarial examples, and 'ResNet-18' means using ResNet-18 to generate black-box adversarial examples

results of ABanditNAS against different white-box attacks for various perturbation budgets on MNIST.

We also compare ABanditNAS with two baselines, UCBNAS and UCBNAS (pruning). Compared with UCBNAS, ABanditNAS can get better performance and use less search time under adversarial attacks as shown in Table 3. Although UCBNAS (pruning) is as fast as ABanditNAS, we can see that it has worse performance than ABanditNAS because of inferior competition among operations before pruning.

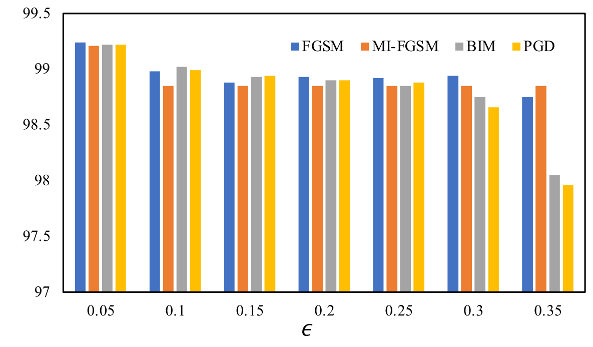
CIFAR-10 We use one Titan V GPU to search, and the entire search process takes about 1.94 hours. We consider $V = 6$ and $V = 10$ cells for training. In ABanditNAS-10, we double the number of normal cells of ABanditNAS-6 to obtain a network with 10 cells, and the reduction cells are located at the 3-th and 6-th cells. In addition, we also train a larger network variant with 100 initial channels for $V = 6$, which denotes as ABanditNAS-6 (channel=100). The results for different architectures on CIFAR-10 are summarized in Tables 4 and 5. Table 4 shows the black-box attacks barely affect the networks obtained by ABanditNAS, much less than those by other methods. On CIFAR-10, more adversarial training methods are compared, such as ADML and LBGAT. The backbone of both methods is ResNet-18. Moreover, it can clearly be seen in Table 5 that ABanditNAS outperforms ResNet, Wide-ResNet and NASNet with higher accuracy under FGSM attack. Compared with Wide-ResNet, ABanditNAS-10 achieves not only a better performance (50.0% vs. 58.74%) in PGD-7, but also fewer parameters (45.9M vs. 5.188M). Although the result of VGG-16 is under PGD-10, ABanditNAS-10 achieves a better performance under a more serious attack PGD-20 (46.04% vs. 50.51%). When compared with NASNet[1] which has a better performance on clean images, our method obtains better performance on adversarial examples with a much faster search

[1] Results are from Cubuk et al. (2017).

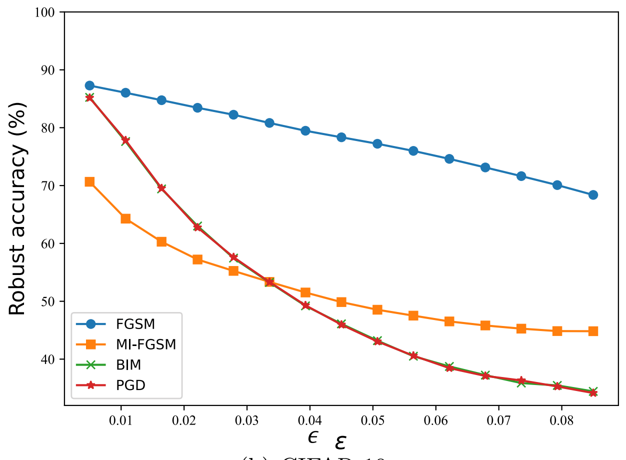
Table 5 Validation accuracy and robustness of various models trained on CIFAR-10.

Architecture	Clean (%)	FGSM (%)	PGD-7 (%)	PGD-20 (%)	# Params (M)	Search Cost (GPU days)	Search Method
VGG-16 Zhang *et al.* (2020)	85.16	46.96	46.04 (PGD-10)	–	–	–	Manual
ResNet Madry et al. (2017)	79.4	51.7	47.1	43.7	12.0	–	Manual
Wide-ResNet Madry et al. (2017)	87.3	56.1	50.0	45.8	45.9	–	Manual
TRADES Zhang et al. (2019)	85.72	53.82	47.98	42.36	12.0	–	Manual
LBGAT Cui et al. (2021)	88.22	–	47.24	42.83	12.0	–	Manual
ADML Yin et al. (2018)	81.32	53.57	48.30	44.12	7.75	–	Manual
NASNet Cubuk et al. (2017)	**93.2**	63.6	50.1	–	3.30	$\sim 7 \times 2000$	RL
DARTS	86.11	55.21	49.72	45.84	4.37	0.83	Gradient
PC-DARTS Xu et al. (2019)	86.43	54.98	49.96	46.22	5.34	0.42	Gradient
ABanditNAS-6	87.16	55.88	50.04	45.90	**2.89**	**0.08**	Anti-Bandit
ABanditNAS-6 (channel=100)	87.31	57.92	51.24	45.79	12.47	**0.08**	Anti-Bandit
ABanditNAS-10	90.64	**81.38**	**58.74**	**50.51**	5.19	**0.08**	Anti-Bandit

Note that the search cost of NASNet which is unknown is estimated based on Cubuk et al. (2017). ‘PGD-10’ means the result of VGG-16 is under PGD-10 attack which comes from Zhang *et al.* (2020). ‘channel=100’ means we initialize the network with 100 channels for V= 6



(a) MNIST



(b) CIFAR-10

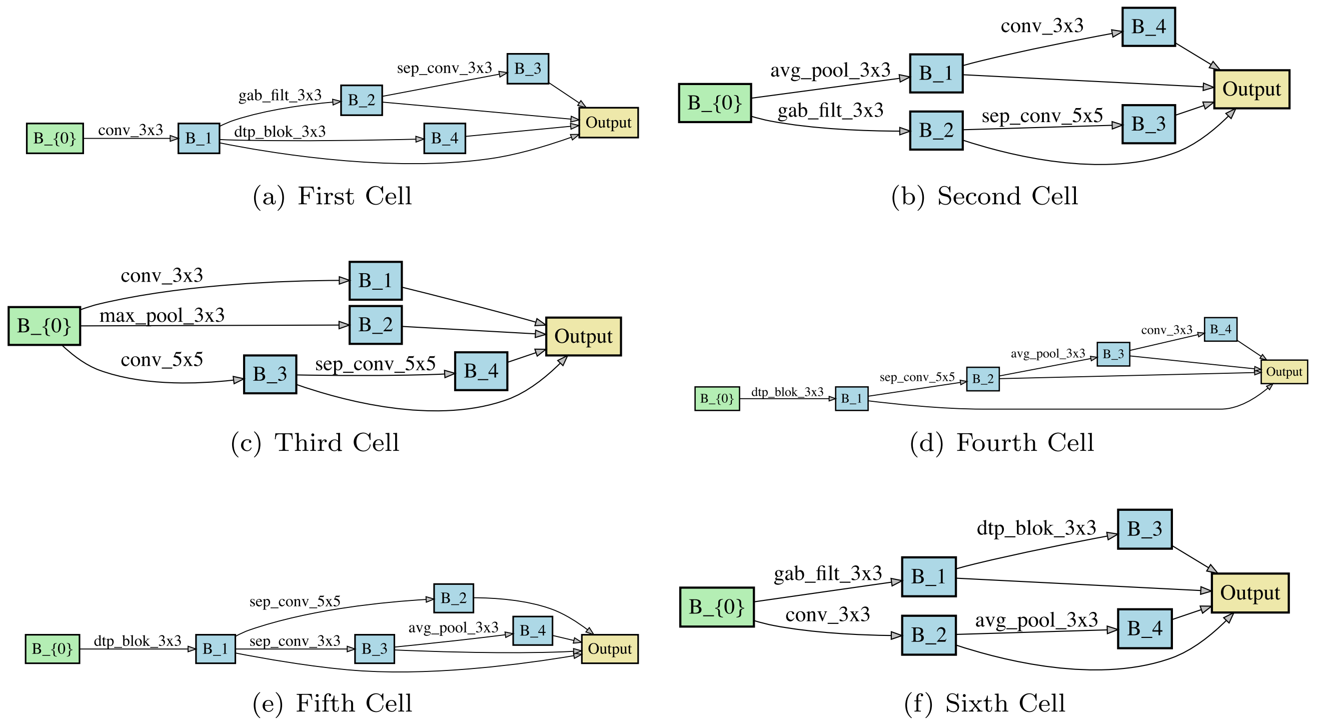
Fig. 5 Performance of ABanditNAS against different white-box attacks for various perturbation budgets

speed ($\sim 7 \times 2000$ vs. 0.08). Also, compared with increasing the initial number of channels, we find that stacking more cells is more effective. Fig. 5b shows more results of ABanditNAS against different attacks for various perturbations. Since BIM and PGD are two similar attack methods, their difference is that PGD randomly initializes and clips its perturbation. After sufficient training, the difference between their curves is marginal. Therefore, their curves almost overlap.

For the structure searched by ABanditNAS on CIFAR-10, we find that the robust structure prefers pooling operations, Gabor filters and denoising blocks (Fig. 6). The reasons lie in that the pooling can enhance the nonlinear modeling capacity, Gabor filters can extract robust features, and the denoising block and mean pooling act as smoothing filters for denoising. Gabor filters and denoising blocks are usually set in the front of the cell by ABanditNAS to denoise features encoded by the previous cell. The setting is consistent with Xie et al. (2019), which demonstrates the rationality of ABanditNAS. Moreover, the robustness of ABanditNAS is verified on MI-FGSM. We compare ABanditNAS with an adversarial method (ADML) and a NAS method (PC-DARTS) on CIFAR-10 as shown in Fig. 7. ABanditNAS outperforms other methods under all perturbation budgets, and the accuracy of ABanditNAS tends to converge with the increase of attack perturbation.

5.4 Analysis

Effect on the hyperparameter λ The hyperparameter λ is used to balance the performance between the past and the current. Different values of λ result in similar search costs. The performances of the structures searched by ABanditNAS with different values of the λ are used to find the best λ. We train the structures in the same setting. From Fig. 8, we can see that when $\lambda = 0.7$, ABanditNAS is most robust.



(a) First Cell (b) Second Cell (c) Third Cell (d) Fourth Cell (e) Fifth Cell (f) Sixth Cell

Fig. 6 Detailed structures of the best cells discovered on CIFAR-10 using FGSM with random initialization

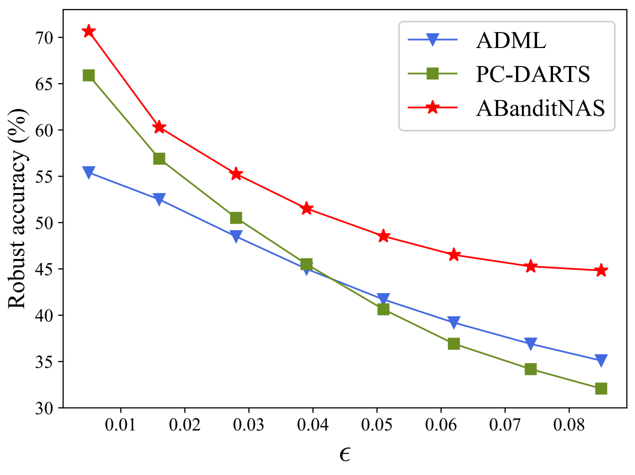


Fig. 7 Robustness of ADML, PC-DARTS and ABanditNAS against MI-FGSM attack for various perturbation budgets

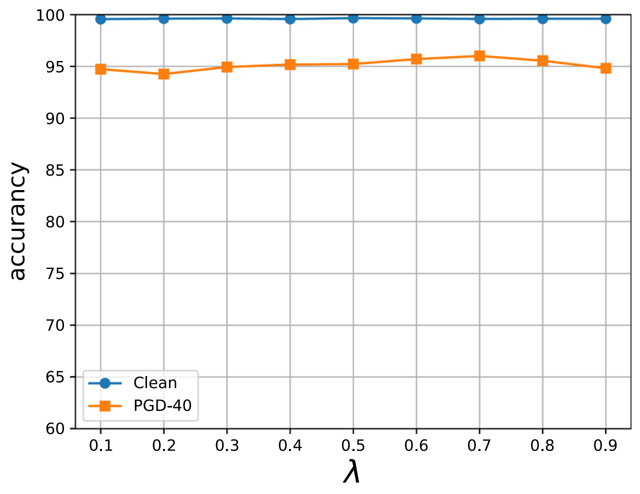


Fig. 8 The performances of the structures searched by ABanditNAS with different values of the hyperparameter λ

Effect on the search space We test the performance of ABanditNAS with different search spaces. In this part, we adopt the same experimental setting as general NAS (as shown in Sec. 5.2). The search space of general NAS has 7 operations. We incrementally add Gabor filter, denoising block, 1×1 dilated convolution with rate 2 and 7×7 dilated convolution with rate 2, until the number of operations in the search space reaches 11. In Table 6, # Search Space represents the number of operations in the search space. Although the difficulty of searching increases with the increase of search space, ABanditNAS can effectively select the appropriate operations. Each additional operation affects little on the search efficiency, which demonstrates the efficiency of our search method. When the number of operations in the search space is 9, the classification accuracy of the model searched by ABanditNAS exceeds all the methods with the same level of search cost.

6 Conclusion

In this paper, we formulate NAS as a multi-bandit problem and introduce ABanditNAS based on LCB and UCB

Table 6 The performance of ABanditNAS with different search spaces on CIFAR10

Architecture	# Search Space	Accuracy (%)	# Params (M)	Search Cost (GPU days)	Search Method
ABanditNAS	7	97.13	3.0	**0.09**	Anti-Bandit
ABanditNAS	8	97.47	3.3	0.11	Anti-Bandit
ABanditNAS	9	97.52	4.1	0.13	Anti-Bandit
ABanditNAS	10	97.53	**2.7**	0.15	Anti-Bandit
ABanditNAS	11	**97.66**	3.7	0.16	Anti-Bandit

to accelerate the training for NAS. We use UCB to prune the operation with the worst performance and LCB to guarantee that every arm is fairly tested. Moreover, based on ABanditNAS, we propose to solve the adversarially robust optimization and architecture search in a unified framework. Experiments on publicly available datasets demonstrate that our ABanditNAS achieves competitive accuracy but with less search cost than other prior methods.

Acknowledgements This work was supported by National Natural Science Foundation of China under Grant 62076016, Beijing Natural Science Foundation L223024, and "One Thousand Plan" innovation leading talent funding projects in Jiangxi Province Jxsg2023102268. Runqi Wang and Linlin Yang are co-first authors. Baochang Zhang is the corresponding author.

Data Availability The datasets analysed during the current study are available in the MNIST LeCun *et al.* (1998), CIFAR-10 Krizhevsky et al. (2009), ImageNet-1k Deng et al. (2009).

Appendix A: Theoretic Analysis

Lemma 1 *The global variance boundary can be represented by the variance boundary of independent trials.*

Proof Suppose X_i represents the estimated value of the operation in i-th trail, which is independently distributed, $X_i \in [0, 1]$, and $X = \frac{\sum_i X_i}{n}$. According to Markov's inequality, for any real-valued variable Y, $P(Y \geq a) \leq E(\frac{Y}{a})$, we can get $P(e^{\lambda X} \geq e^{\lambda a}) \leq \frac{E(e^{\lambda X})}{e^{\lambda a}}$, where λ is a constant. n is the number of times of the arm has been played up to trial. We can get Eq. 15 using Jeason inequality and AM-GM inequality.

$$\begin{aligned} E(e^{\lambda X}) &= E(e^{\frac{\lambda}{n}\sum_i X_i}) \\ &= \prod_i (e^{\frac{\lambda}{n}X_i}) \\ &\leq \prod_i (X_i e^{\frac{\lambda}{n}} + q_i) \\ &\leq (\frac{\sum_i (X_i e^{\frac{\lambda}{n}} + q_i)}{n})^n \\ &= (Xe^{\frac{\lambda}{n}} + q)^n \end{aligned} \quad . \tag{15}$$

Here, $q_i = 1 - X_i$ and $q = 1 - \frac{\sum_i X_i}{n}$. □

References

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

Bender, G., Kindermans, P.-J., Zoph, B., Vasudevan, V.& Le, Q.V. (2018). Understanding and simplifying one-shot architecture search. In: *The international conference on machine learning* .

Buades, A., Coll, B. & Morel, J.-M. (2005). A non-local algorithm for image denoising. In: *The IEEE / CVF computer vision and pattern recognition conference*.

Cai, H., Chen, T., Zhang, W., Yu, Y. & Wang, J. (2018). Efficient architecture search by network transformation. In: *The association for the advancement of artificial intelligence* .

Cai, H., Zhu, L.& Han, S.(2019). ProxylessNAS: Direct neural architecture search on target task and hardware. In: *The international conference on learning representations*.

Carlini, N.& Wagner, D. (2017). Towards evaluating the robustness of neural networks. In: *IEEE Symposium on Security and Privacy*.

Chen, Xin, Xie, Lingxi, Wu, Jun & Tian, Qi (2019). Progressive differentiable architecture search: Bridging the depth gap between search and evaluation. In ICCV.

Chen, X.& Hsieh, C.-J.(2020). Stabilizing differentiable architecture search via perturbation-based regularization. In: *The international conference on machine learning*.

Chen, H., Zhang, B., Xue, S., Gong, X., Liu, H., Ji, R.& Doermann, D. (2020). Anti-bandit neural architecture search for model defense. In: *The european conference on computer vision*.

Cisse, M., Bojanowski, P., Grave, E., Dauphin, Y. & Usunier, N. (2017). Parseval networks: Improving robustness to adversarial examples. In: *The international conference on machine learning*.

Cubuk, E.D., Zoph, B., Schoenholz, S.S. & Le, Q.V. (2017). Intriguing properties of adversarial examples. In *The international conference on learning representations*.

Cui, J., Liu, S., Wang, L.& Jia, J.(2021). Learnable boundary guided adversarial training. In: *The international conference on computer vision*.

Dapello, J., Marques, T., Schrimpf, M., Geiger, F., Cox, D.& DiCarlo, J.J. (2020). Simulating a primary visual cortex at the front of cnns improves robustness to image perturbations. In NeurIPS.

Das, N., Shanbhogue, M., Chen, S.-T., Hohman, F., Chen, L., Kounavis, M.E. & Chau, D.H.(2017). Keeping the bad guys out: Protecting and vaccinating deep learning with jpeg compression. arXiv:1705.02900.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K.& Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In: *The IEEE / CVF computer vision and pattern recognition conference*.

DeVries, T.& Taylor, G.W. (2017). Improved regularization of convolutional neural networks with cutout. arXiv:1708.04552.

Dong, N., Xu, M., Liang, X., Jiang, Y., Dai, W.& Xing, E.(2019). Neural architecture search for adversarial medical image segmentation. In: *Medical image computing and computer assisted intervention*.

Dziugaite, G.K. , Ghahramani, Z.& Roy, D.M. (2016). A study of the effect of jpg compression on adversarial images. arXiv:1608.00853.

Even-Dar, E., Mannor, S., & Mansour, Y. (2006). Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems. *Journal of Machine Learning Research, 7*(39), 1079–1105.

Gabor, D. (1946). Theory of communication. part 1: The analysis of information. *Journal of the Institution of Electrical Engineers-Part III: Radio and Communication Engineering, 93*(26), 429–441.

Gabor, D. (1946). Electrical engineers-part III: Radio and communication engineering. *Journal of the Institution of Electrical Engineers - Part III: Radio and Communication Engineering, 93*(429), 39.

Gavin, A. R. and Mahesan, N. (1994) On-line Q-learning using connectionist systems, volume 37. University of Cambridge, Department of Engineering Cambridge.

Goodfellow, I.J., Shlens, J.& Szegedy, C. (2015). Explaining and harnessing adversarial examples. In: *The international conference on learning representations*.

Guo, M., Yang, Y., Xu, R., Liu, Z.& Lin, D. (2020). When nas meets robustness: In search of robust architectures against adversarial attacks. In: *The IEEE / CVF computer vision and pattern recognition conference*.

Gupta, P.& Rahtu, E. (2019). Defeating adversarial attacks by fusing class-specific image inpainting and image denoising: Ciidefence. In: *The international conference on computer vision*.

He, K., Zhang, X., Ren, S., Sun, J.(2016). Deep residual learning for image recognition. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., & Adam, H. (2017). Mobilenets Efficient convolutional neural networks for mobile vision applications. *Transactions on Image Processing, 30*, 1291–1304.

Huang, G., Liu, Z., Van Der Maaten, L.& Weinberger, K.Q. (2017). Densely connected convolutional networks. In: *The IEEE / CVF computer vision and pattern recognition conference*.

Ilyas, A., Engstrom, L.& Madry, A.(2018). Prior convictions: Black-box adversarial attacks with bandits and priors. In: *The international conference on learning representations*.

Kotyan, S.& Vargas, D.V.(2020). Evolving robust neural architectures to defend from adversarial attacks. In: *CEUR Workshop*.

Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images. In: *Citeseer*.

Kurakin, A., Goodfellow, I.J. & Bengio, S. (2016). Adversarial examples in the physical world. In: *The international conference on learning representations*.

Lai, T. L., Robbins, H., et al. (1985). Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics, 6*(1), 4–22.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE, 86*(11), 2278–2324.

Li, G., Qian, G., Delgadillo, I.C., Muller, M., Thabet, A.& Ghanem, B.(2020). Sgas: Sequential greedy architecture search. In: *The IEEE / CVF computer vision and pattern recognition conference* .

Liao, F., Liang, M., Dong, Y., Pang, T., Hu, X.& Zhu, J. (2018). Defense against adversarial attacks using high-level representation guided denoiser. In: *International conference on pattern recognition*.

Liu, Y., Chen, X., Liu, C.& Song, D.(2016). Delving into transferable adversarial examples and black-box attacks. In: *The international conference on learning representations*.

Liu, H., Simonyan, K.& Yang, Y.(2018). Darts: Differentiable architecture search. In: *The international conference on learning representations*.

Liu, C., Zoph, B., Neumann, M., Shlens, J., Hua, W., Li, L.-J., Fei-Fei, L., Yuille, A., Huang, J.& Murphy, K.(2018). Progressive neural architecture search. In: *The European conference on computer vision*.

Long, J., Shelhamer, E., Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*.

Ma, N., Zhang, X., Zheng, H.-T. & Sun, J.(2018). Shufflenet v2: Practical guidelines for efficient cnn architecture design. In: *The European conference on computer vision*.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D.& Vladu, A.(2017). Towards deep learning models resistant to adversarial attacks. In: *The international conference on learning representations*.

Na, T., Ko, J.H.& Mukhopadhyay, S. (2017). Cascade adversarial machine learning regularized with a unified embedding. In: *The international conference on learning representations*.

Osadchy, M., Hernandez-Castro, J., Gibson, S., Dunkelman, O., & Pérez-Cabo, D. (2017). No bot expects the deepcaptcha! introducing immutable adversarial examples, with applications to captcha generation. *IEEE Transactions on Information Forensics and Security, 12*(11), 2640–2653.

Pérez, J.C., Alfarra, M., Jeanneret, G., Bibi, A., Thabet, A.K., Ghanem, B.& Arbeláez, P.(2020). Gabor layers enhance network robustness. In: *The European conference on computer vision*.

Pham, H., Guan, M., Zoph, B., Le, Q.& Dean, J. (2018). Efficient neural architecture search via parameter sharing. In: *The international conference on machine learning*.

Pinto, A.S., Kolesnikov, A. , Shi, Y., Beyer, L.& Zhai, X. (2023). Tuning computer vision models with task rewards. arXiv-2302.

Real, E., Aggarwal, A., Huang, Y.& Le, Q.V. (2018). Regularized evolution for image classifier architecture search. In: *The association for the advancement of artificial intelligence*.

Samangouei, P., Kabkab, M.& Chellappa, R. (2018). Defense-GAN: Protecting classifiers against adversarial attacks using generative models. In *The international conference on learning representations*.

Sheth, P.& Xie, P. (2023). Improving differentiable neural architecture search by encouraging transferability. In: *The international conference on learning representations*.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. (2017). Mastering the game of go without human knowledge. *Nature, 550*(7676), 354–359.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V.& Rabinovich, A.(2015). Going deeper with convolutions. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V.& Rabinovich, A.(2015). Going deeper with convolutions. In: *The IEEE / CVF computer vision and pattern recognition conference*.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I.& Fergus, R.(2013). Intriguing properties of neural networks. In: *The international conference on learning representations*.

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

Xie, C., Wu, Y., van der Maaten, L., Yuille, A.L. & He, K. (2019). Feature denoising for improving adversarial robustness. In: *International conference on pattern recognition*.

Xie, S., Zheng, H., Liu, C.& Lin, L.(2018). Snas: stochastic neural architecture search. In: *The international conference on learning representations*.

Xu, Y., Xie, L., Zhang, X., Chen, X., Qi, G.-J., Tian, Q.& Xiong, H.(2019). Pc-darts: Partial channel connections for memory-efficient differentiable architecture search. In: *The international conference on learning representations*.

Xue, S., Wang, R., Zhang, B., Wang, T., Guo, G.& Doermann, D. (2021). Idarts: Interactive differentiable architecture search. In: *The international conference on computer vision*.

Yang, Y., Zhang, G., Katabi, D.& Xu, Z. (2019). Me-net: Towards effective adversarial robustness with matrix estimation. In: *The international conference on machine learning*.

Yin, C., Tang, J., Xu, Z.& Wang, Y.(2018). Adversarial meta-learning. arXiv:1806.03316.

Ying, C., Klein, A., Christiansen, E., Real, E., Murphy, K.& Hutter, F.(2019). Nas-bench-101: Towards reproducible neural architecture search. In: *The international conference on machine learning*.

Zhang, Z., Wang, X., Guan, C., Zhang, Z., Li, H.& Zhu, W.(2023). Autogt: Automated graph transformer architecture search. In: *The international conference on learning representations*.

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

Zhang, X., Zhou, X., Lin, M.& Sun, J.(2018). Shufflenet: An extremely efficient convolutional neural network for mobile devices. In: *The IEEE / CVF computer vision and pattern recognition conference* .

Zhang, C., Liu, A., Liu, X., Yitao, X., Hang, Yu., Ma, Y., & Li, T. (2020). Interpreting and improving adversarial robustness of deep neural networks with neuron sensitivity. *Transactions on Image Processing, 30*, 1291–1304.

Zheng, X., Ji, R., Tang, L., Wan, Y., Zhang, B., Wu, Y., Wu, Y.& Shao, L.(2019). Dynamic distribution pruning for efficient network architecture search. CoRR, arXiv:1905.13543.

Zhou, H., Chen, K., Zhang, W., Fang, H., Zhou, W.& Yu, N.(2019). Dup-net: Denoiser and upsampler network for 3d adversarial point clouds defense. In: *The international conference on computer vision*.

Zhou, J., Zheng, L., Wang, Y., Wang, C., & Gao, R. X. (2022). Automated model generation for machinery fault diagnosis based on reinforcement learning and neural architecture search. *IEEE Transactions on Instrumentation and Measurement, 71*, 1–12.

Zoph, B.& Le, Q.V. (2016). Neural architecture search with reinforcement learning. In: *The international conference on learning representations*.

Zoph, B., Vasudevan, V., Shlens, J.& Le, Q.V. (2018). Learning transferable architectures for scalable image recognition. In: *The IEEE / CVF computer vision and pattern recognition conference* .

Zoph, B., Vasudevan, V., Shlens, J.& Le, Q.V. (2018). Learning transferable architectures for scalable image recognition. In: *The IEEE / CVF computer vision and pattern recognition conference*..