A Survey on Efficient Methods for Adversarial Robustness

AWAIS MUHAMMAD AND SUNG-HO BAE, (Member, IEEE)
Department of Computer Science and Engineering, Kyung-Hee University, Seoul 17104, South Korea
Corresponding author: Sung-Ho Bae (shbae@khu.ac.kr)

This work was supported in part by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT (MSIT) and Future Planning under Grant 2018R1C1B3008159, in part by the Institute of Information and Communications Technology Planning and Evaluation (IITP) funded by the Korean Government (MSIT) through Artificial Intelligence Innovation Hub under Grant 2021-0-02068, in part by the Institute of Information & Communications Technology Planning & Evaluation (IITP) funded by the Korea Government (MSIT) under Grant RS-2022-00155911, and in part by the Artificial Intelligence Convergence Innovation Human Resources Development through Kyung Hee University.

ABSTRACT Deep learning has revolutionized computer vision with phenomenal success and widespread applications. Despite impressive results in complex problems, neural networks are susceptible to adversarial attacks: small and imperceptible changes in input space that lead these models to incorrect outputs. Adversarial attacks have raised serious concerns, and robustness to these attacks has become a vital issue. Adversarial training, a min-max optimization approach, has shown promise against these attacks. The computational cost of adversarial training, however, makes it prohibitively difficult to scale as well as to be useful in practice. Recently, several works have explored different approaches to make adversarial training computationally more affordable. This paper presents a comprehensive survey on efficient adversarial robustness methods with an aim to present a holistic outlook to make future exploration more systematic and exhaustive.

INDEX TERMS Neural networks, deep learning, efficient training, adversarial robustness, adversarial attacks, adversarial learning.

I. INTRODUCTION Deep learning methods have shown exemplary performance on a multitude of tasks in computer vision, such as image recognition, object detection, semantic segmentation, etc. Unfortunately, these models are also vulnerable to small and imperceptible changes in the input space called adversarial examples. Adversarial examples are tiny changes in the input that can fool even state-of-the-art neural networks. This vulnerability of neural networks has raised grave security and robustness concerns. The robustness of models is not only essential to defend against malicious adversaries but also brings desirable...
properties [24], [42], [71], [72], [81], [83], [102]. It has been shown that robust models are easier to interpret [24], [72], learn better and more transferable features [42], [71], [83], are better priors for image generation [24], [72], [81], [102], etc. Many robustness methods have been proposed in recent years to defend against adversarial perturbations. These methods include defensive distillation which uses distillation [65], model compression [17], [55], activation pruning [19], [66], gradient regularization [30], [63], [70], better layer utilization [5] and adversarial training [57], etc. But adversarial training is one general method that has stood the test of time [4]. Adversarial training consists of the generation of adversarial examples. In recent years, many variants of adversarial training have been proposed, but they follow the main principle defined by adversarial training.

Adversarial training formulates robustness as an optimization problem that requires the generation of optimal adversarial examples and then training the model on these examples [57]. However, finding an optimal adversarial example is an NP-hard problem [44], [86]. For this reason, adversarial training methods utilizes approximation methods to find adversarial examples. The first successful adversarial training method, projected gradient descent-based adversarial training (PGD-AT), uses iterative optimization to find adversarial examples. It consists of an inner, iterative maximization step to find the adversarial examples. This step utilizes gradient w.r.t. input and requires several steps to converge. Then, the outer step minimizes the loss to find the optimal parameters. Several improvements to this method have been proposed [85], [94] but all of them are based on the same general principle.

Adversarial training is more challenging as it requires significantly larger training time to converge compared with normal training [57], [73], [87]. For instance, PGD-7 takes seven times more time compared with normal training. The training time becomes prohibitively large at scale. For instance, a model that takes a week to train normally may require 50 days for adversarial training [88]. We provide a comparison of normal training and adversarial training in Figure 1 taken from Zheng et al. [100]. It is crucial to make adversarial training more efficient in order to utilize it for practical purposes.

Since the introduction of adversarial training [57], several interesting methods have been proposed to instill robustness more efficiently [6], [7], [73], [87], [93]. For instance, Shafahi et al. [73] proposed to recycle gradients for the adversary generation. Similarly, Zhang et al. [93] observed that only the first layer has a significant link to the adversary update and hence proposed to use gradients of the only first layer for adversary update. Wong et al. [87] proposed a method that can achieve robustness by a single inner maximization step making the original adversarial training much faster. Furthermore, Shafahi et al. [74] proposed to transfer robustness from a robust pre-trained model, and Awais et al. [6] proposed to distill robustness from intermediate activations of a robust teacher.

In this paper, we present a holistic overview of efficient methods for adversarial robustness. Our aim is to provide a comprehensive overview of these methods that can be utilized as a guide for future developments in the field. We start by defining the mathematical foundations of adversarial attacks and adversarial robustness. We then presented a new organization of efficient adversarial training methods. This organization is based on the underlying approach used by these methods. Then, we presented a comprehensive overview of these methods along with a comparative analysis of their performance. Finally, we also discuss possible future directions that can help improve the efficiency and performance of these methods. Concisely, the goals of this paper are as follows:

- To present a comprehensive overview of adversarial robustness methods that could be used as a guide for the future development
- To provide an exhaustive comparison of the performance of efficient adversarial robustness methods to act as a baseline for efficient robustness studies
- To offer a thorough discussion of the limitation of existing works and possible future directions

II. FOUNDATIONS

In this section, we first provide a foundation for the rest of the paper. We start by formulating the robustness problem mathematically. Then, we explain adversarial attacks and adversarial training.

A. PROBLEM FORMULATION

Consider the task of mapping $x \in \mathcal{X} \subseteq \mathbb{R}^d$ to $y \in \mathcal{Y} \subseteq \{1, 2, ..., K\}$, where $K$ is number of classes. Given training data $\mathcal{D} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, the goal is to learn a model $f_\theta : \mathcal{X} \rightarrow \mathcal{Y}$ parameterized by $\theta$ from a hypothesis space $\mathcal{F}$.

\begin{align*}
\mathcal{X} & \quad \quad \delta \\
\rightarrow & \quad \quad = \\
\mathcal{X}_{adv} & \quad \quad (\text{Prediction: Cat (54%)})
\end{align*}

\begin{align*}
\mathcal{X} & \quad \quad \delta \\
\rightarrow & \quad \quad = \\
\mathcal{X}_{adv} & \quad \quad (\text{Prediction: Stop (45%)})
\end{align*}

\begin{align*}
\mathcal{X} & \quad \quad \delta \\
\rightarrow & \quad \quad = \\
\mathcal{X}_{adv} & \quad \quad (\text{Prediction: Speed Limit (99%)})
\end{align*}

FIGURE 2. An illustration of adversarial examples. A small, imperceptible perturbation added to the input space can change the prediction of an otherwise accurate neural network. The first example demonstrates a cat image being converted to a dog with the addition of $\epsilon$-small and imperceptible noise. The second part illustrates a more serious attack where a stop sign is converted into a speed limit sign by the addition of a small perturbation.
This goal is accomplished by minimizing a loss function $\mathcal{L}(f(x), y)$ on training data $D$. A model is said to be robust if $\forall \|\delta\|_p \leq \epsilon$, $f(x + \delta) = f(x)$, where $p$ is a norm ($p \in \{1, 2, \infty\}$), $\epsilon$ is budget for the perturbation and $\delta$ is perturbation that is added to original example to get the adversarial example ($x_{adv} = x + \delta$). In practice, the robustness is evaluated on the left-out validation set $D_{val}$ by constructing adversarial attacks detailed in the next section.

**B. ADVERSARIAL ATTACKS**

Despite huge success, Szegedy et al. [79] discovered an intriguing weakness of neural networks. They showed that these models are susceptible to small adversarial perturbations. It is possible to fool these models by adding tiny and imperceptible perturbations in the input space that are not even visible to the human visual system. These perturbations not only fool models but also increase their confidence in the model on the wrong assumptions. A demonstration of adversarial perturbation is presented in Figure 2. This discovery sparked a strong interest in researchers, and a plethora of attacks have been introduced ever since [22], [29], [31], [51], [62], [63], [64]. In this section, we provide a brief overview of two attacks that are relevant to adversarial robustness. We refer the reader to Akhtar and Mian [1] for a comprehensive treatment of the subject.

Generally, finding an attack is formulated as a constrained optimization problem defined as follows.

$$
\delta^* = \arg \max_{\delta} \mathcal{L}(f_{\theta}(x + \delta), y) \quad \text{s.t.} \quad \|\delta\|_p \leq \epsilon. \quad (1)
$$

However, finding an optimal perturbation in an NP-hard problem [44], [86]. Hence, the perturbation is approximated with gradient descent. This first method, Fast Gradient Sign Method (FGSM) [29], solves the optimization based on the local linear approximation of loss and is defined as follows.

$$
\delta_{\text{FGSM}} = \epsilon \cdot \text{sgn}(\nabla_x \mathcal{L}(x, y; \theta)). \quad (2)
$$

However, the success rate of FGSM attacks is limited. It is possible to construct a better approximation by taking multiple smaller steps with gradient ascent. This type of iterative attack is called projected gradient descent (PGD). This PGD-$k$ attack can be defined as follows.

$$
\delta_{\text{PGD}} = \Pi_{(-\epsilon, \epsilon)}[\delta^{k-1} + \alpha \cdot \text{sgn}(\nabla_x \mathcal{L}(x + \delta^k, y; \theta))], \quad (3)
$$

where $\Pi$ is the projection operator and $k$ is the number of steps for gradient ascent.

**C. ADVERSARIAL TRAINING**

The adversarial training (AT) tackles the problem of making models robust. The main idea behind adversarial training is to create strong, iterative adversarial perturbations during the training and train the model on these perturbed inputs.
This goal is achieved by min-max optimization [57]. The inner maximization part consists of constructing strong adversarial perturbations and the outer-loop consists of minimization to find optimal $\theta$. The adversarial training can be formulated as following min-max problem.

$$\min_{\theta} \sum_{i=0}^{n} \max_{\|\delta\| \leq \epsilon} \mathcal{L}(x_i + \delta, y_i, \theta).$$

The inner-maximization problem is solved by Equation 3. We refer to this method as adversarial training or AT in the rest of the paper. Adversarial training utilizes PGD-7 attacks for its training. Note that the computational complexity of adversarial attack for AT is $O(k.p.b)$ where $k$ is the number of steps for inner maximization in Equation 4, $p$ is the cost of forward-propagation steps, and $b$ is the cost of backward-propagation step.

**D. STUDY SELECTION METHODOLOGY**

1) SELECTION CRITERIA

This study aims to understand and report recent advances in efficient adversarial robustness methods and to present a holistic overview of the area that can be utilized for future research. To fulfill this purpose, we selected the following sources for the search for relevant articles: SCOPUS, Web of Sciences, and Google Scholar. To get relevant results, we performed a search with the following search terms: ‘efficient adversarial training deep learning robustness cifar10 time’.

2) EXCLUSION AND INCLUSION CRITERIA

To get the relevant articles, we devised simple inclusion and exclusion criteria. This criterion consists of the following filters. The paper should have appeared after the publication of the original adversarial training paper [57] and before June 2022. It should be published in a reputable venue. To accomplish this, we searched for the venue of selected articles on the Web of Sciences and the ranking provided by Google Scholar. We also filtered the papers based on the reporting of results. Since reporting PGD-based robustness on CIFAR10 is critical for a fair comparison, we made it a mandatory criterion. As a final consideration, we made sure the study aimed at improving adversarial training efficiency.

3) SELECTION RESULTS

To select relevant articles, we performed a search based on the above-mentioned search terms and criteria in June. This search returned a total of 9170 results. To choose the most relevant studies, we utilized a four-step filtering strategy. We started by reading the titles and removing clearly irrelevant articles. We then proceeded to abstracts and read them to make sure that study is focused on efficient robustness and provides results on relevant datasets. Then, we skimmed through the introduction and experimental results. Finally, to further ensure coverage of relevant works, we examined citations and related works of the selected papers as well. The whole process leads us to a total of 27 articles. We are left with 21 articles after reading and extracting results and methodologies thoroughly.

**III. EFFICIENT ADVERSARIAL ROBUSTNESS METHODS**

We broadly categorized efficient adversarial training methods into two classes: methods that modify original adversarial training (defined in Equation 4) to improve its efficiency and methods that transfer robustness from pre-trained models. An overview of this categorization is depicted in Figure 3. We also provide a list of papers under each category in Table 1. First, in section III-A, we discussed methods that modify the inner loop of the adversarial training method. These methods generally exploit different inefficiencies in the original formulation of adversarial training. Then, in Section III-B, we discuss approaches that employ different transfer learning strategies to make adversarial learning more affordable. These methods leverage different transfer learning approaches to transfer the robustness across models and datasets.

A. EFFICIENT ADVERSARIAL TRAINING

Adversarial training relies on generating adversarial examples during training. However, finding optimal adversarial examples is an NP-hard problem [44], [86]. In adversarial training, adversarial examples are approximated by iterative projected gradient descent (PGD) [57].

The multi-step nature of PGD makes the training expensive and harder to scale. For instance, PGD-7-based adversarial training [57] requires $7 \times$ more compute power compared with normal training. In this section, we discuss methods proposed to make the PGD-$k$ efficient. We have classified these methods into three sub-categories: methods that employ variants of single-step attack methods to approximate adversarial perturbations, methods that recycle gradients, and techniques that utilize partial training. We discuss these categories one by one. A quantitative comparison of the results of these methods is given in Table 2 and Figure 4.

1) SINGLE-STEP ADVERSARIAL TRAINING METHODS

The adversarial training methods can be categorized based on their approach to approximating adversarial perturbation (inner maximization loop in Equation 4). Adversarial training [57] uses a multi-step approach, which results in a better but more expensive approximation. Single-step adversarial attacks, on the other hand, approximate perturbations with a single step of gradient descent. Single-step adversarial attacks, such as Fast Gradient Sing Method (FGSM) [29], were shown to be ineffective against multi-step adversarial attacks. This led to the formulation of multi-step PGD attack-based training [57].

However, Wong et al. [87] has recently shown that the reason behind the failure of FGSM-based adversarial training is catastrophic overfitting. The catastrophic overfitting is a phenomenon whereby FGSM-based training leads to robustness in the early phase of the training, but it loses all of
its robustness in a single epoch. The catastrophic overfitting is demonstrated in Figure 5. The authors showed that it is possible to mitigate catastrophic overfitting by initialization the attack by adding random noise in the perturbation. The adversarial attack for Fast Adversarial Training (FastAT) can be formulated as follows.

$$\delta = \Pi_{(-\epsilon, \epsilon)} [\eta + \epsilon \cdot \text{sgn}(\nabla_x \mathcal{L}(x, y, \theta))]$$

where $\eta \sim \mathcal{U}([-\epsilon, \epsilon]^d)$ is $d$-dimensional noise sampled from uniform distribution, $d$ is the dimension of inputs, and $\Pi$ is the projection operator.

FastAT also introduced a few other parameters to further improve the efficiency and results. These extensions include early stopping, larger step size ($\alpha$), mixed precision arithmetic [59] and use of cyclic learning rates [77]. All of these extensions along with a single-step attack formulation makes FastAT computationally significantly less expensive. The time complexity for inner-loop of this algorithm is $O(p, b)$ as it takes extra forward ($p$) and backward ($b$) step.

Although the addition of noise solves the catastrophic overfitting problem, there is no clear explanation for why the addition of noise is helpful. The Andriushchenko and Flammarion [3] first investigated the effectiveness of random noise for the mitigation of catastrophic overfitting. The empirically showed that not only FGSM but both Fast [87] and Free [73] adversarial training suffer catastrophic overfitting. Moreover, they theoretically showed that the addition of random noise decreases the expected magnitude of the perturbation. The smaller magnitude perturbations ($\|\delta\|_2$) benefit from a better linear approximation. Recall that the FGSM is a close-form solution of the adversarial training objective function.

Furthermore, they established a connection between overfitting and local linearity. To this end, they defined a local linearity metric called gradient alignment. Based on this metric, they showed that catastrophic overfitting happens in the later stages of training when SGD starts to learn a model of increasing complexity thereby reducing the linearity of the objective function. Furthermore, they showed that in extreme cases, even simple models (single filter models) can also cause this type of catastrophic overfitting.

| Method Name         | Paper Name                                                                 |
|---------------------|----------------------------------------------------------------------------|
| FastAT              | Fast is better than free: Revisiting adversarial training [87]             |
| GradAlign           | Understanding and Improving Fast Adversarial Training [32]                 |
| FreeBAT             | Revisiting and Advancing Fast Adversarial Training Through The Lens of Bi-Level Optimization [95] |
| Backward Smoothing  | Efficient Robust Training via Backward Smoothing [15]                     |
| N-FGSM              | Make Some Noise: Reliable and Efficient Single-Step Adversarial Training [18] |
|                     | Adversarial Training with Recycling of Gradient Information                |
| FreeAT              | Adversarial Training for Free! [73]                                        |
| ATTA                | Efficient Adversarial Training With Transferable Adversarial Examples [100] |
|                     | Partial Adversarial Training Methods                                       |
| YOPO                | You Only Propagate Once: Accelerating Adversarial Training via Maximal Principle [93] |
| DelayedAT           | Improving the affordability of robustness training for DNNs [32]           |
| BulletTrain         | BulletTrain: Accelerating Robust Neural Network Training via Boundary Example [39] |
| CoresetAT           | $\ell_\infty$-Robustness and Beyond: Unleashing Efficient Adversarial Training [20] |
|                     | Transfer and Distillation                                                  |
| RobustPreTrain      | Using pre-training can improve model robustness and uncertainty [34]       |
| Rob. TL             | Adversarially robust transfer learning [74]                                |
|                     | Labels-based Robustness Distillation                                       |
| Adv. Distillation   | Adversarially Robust Distillation [27]                                     |
| RSALAD              | Revisiting Adversarial Robustness Distillation: Robust Soft Labels Make Student Better [103] |
| AKD                 | On the benefits of knowledge distillation for adversarial robustness [58]   |
|                     | Input Gradient-based Robustness Distillation                              |
| IGAM                | What it Thinks is Important is Important: Robustness Transfers through Input Gradients [14] |
| KDIGA               | How and When Adversarial Robustness Transfers in Knowledge Distillation? [75] |
|                     | Intermediate Activations-based Robustness Distillation                    |
| MixACM              | MixACM: Mix-up-Based Robustness Transfer via Distillation of Activated Channel Maps [6] |
| Attention Transfer  | Adversarial robustness via attention transfer [54]                        |
| RRM                 | Transferring Adversarial Robustness Through Robust Representation Matching [84] |

TABLE 1. A list of papers on efficient adversarial training along with corresponding abbreviations for the proposed methods.
Finally, they proposed to use gradient alignment as a regularizer during FGSM training. This, they suggested, can improve robustness and avoid catastrophic overfitting. The main idea of GradAlign-based adversarial training is to maximize gradient alignment between the gradients at a point at $x$ and randomly perturbed point $x + \eta$ inside the $\ell_\infty$-norm ball around $x$. For this, they propose to add the following alignment regularizer in the training objective of FastAT.

$$\Omega (x, y, \theta) = \mathbb{E}_{(x, y)} \sim D, \eta \sim U([-\eta, \eta]^d)[1 - \cos (\nabla_x \ell (x, y; \theta), \nabla_x \ell (x + \eta, y; \theta))] .$$

Note that this regularizer requires double back-propagation, making the method more expensive compared with FastAT. The time complexity for the inner-loop of this algorithm is $O(2.p.b)$ as it takes two extra forward ($p$) and backward ($b$) steps.

Similar to GradAlign, Chen et al. [15] also tried to understand the contribution of random initialization to the success of FastAT. They conjectured that random initialization makes the objective function of FastAT smoother. The smoothness of an objective function is directly related to the maximum step size that can be taken in gradient-descent with guaranteed convergence. FastAT, unlike PGDAT, requires large step size as $\epsilon = \alpha$. The use of large step size is important in FastAT as it has only one step and the adversarial attack needs to be strong. Based on these observations, they hypothesized that randomized smoothing is the main reason behind the success of FastAT.

Based on this randomized smoothing hypothesis, they also proposed a method that uses randomized smoothing to further improve the FastAT. However, their method does not utilize randomized smoothing in the input space but in the output space. In short, they aim to find optimal input perturbation $\xi$ such that: $f_0 (x + \xi) = f_0 (x) + \gamma \eta$. 

| Method          | Model               | Inner-loop Time Complexity | Clean Accuracy | Robustness |
|-----------------|---------------------|----------------------------|----------------|------------|
| AT [57]         | ResNet50            | $O(k.p.b)$                 | 87.30%         | 45.80%     |
| FastAT [87]     | ResNet18            | $O(p.b)$                   | 83.81%         | 46.06%     |
| GradAlignAT [3] | ResNet18            | $O(2.p.b)$                 | 81.00%         | 47.58%     |
| Backward Smoothing [15] | ResNet18 | $O(2.p.b)$ | 82.38% | 52.50% |
| N-FGSM [18]     | ResNet18            | $O(p.b)$                   | 80.80%         | 48.48%     |
| FreeAT [73]     | ResNet50            | $O(p.b)$                   | 85.96%         | 46.33%     |
| ATTA [100]      | WideResNet34-10     | $O(k.p.b)/r$               | 85.71%         | 50.96%     |
| YOPO [93]       | WideResNet34-10     | $O(k.p.b/l)$               | 83.99%         | 44.72%     |
| DelayedAT [32]  | WideResNet28-10     | -                          | 87.90%         | 53.60%     |
| CrosssetAT [20] | ResNet50            | -                          | 81.65%         | 47.04%     |
| BulletTrainAT [39]| WideResNet34-10   | -                          | 85.89%         | 56.35%     |
A. Muhammad, S.-H. Bae: Survey on Efficient Methods for Adversarial Robustness

FIGURE 6. A visual comparison of FGSM [29], RS-FGSM (used in FastAT) [87] and N-FGSM [18] attacks. FGSM finds a perturbation inside of $\epsilon$-defined norm-ball. RS-FGSM adds random noise in FGSM perturbation and then projects the perturbation back to the $\epsilon$-bounded norm ball. N-FGSM uses a larger noise step and skips the projection. This figure is based on Jorge et al. [18].

In order to find the optimal perturbation in the input space, they optimized the following function:

$$\xi^* = \arg \min_{\xi \in \mathbb{R}} KL(f_\theta(x) + \gamma \psi(x + \xi))$$

This optimal input perturbation is then added in perturbation (e.g., $\delta + \xi^*$) for the inner-maximization loop of the adversarial training method. The time complexity for the inner-loop of this algorithm is similar to GradAlign: $O(2,p,b)$ as it takes two extra forward ($p$) and backward ($b$) steps.

The de Jorge et al. [18] revisited the importance of noise in FGSM-based adversarial training. Contrary to Andriushchenko et al. [2], they empirically showed that increasing the magnitude of the noise is important to prevent catastrophic overfitting. Based on this observation and their experimental findings, they proposed to eliminate the projection (also called clipping) step in Equation 5. They argued that the projection step reduces the effectiveness of a single-step adversary. Based on these, the proposed Noisy-FGSM (N-FGSM) can be formulated as follows:

$$\delta_{N-FGSM} = \eta + \epsilon \cdot \text{sgn}(\delta_x L_\theta(x_i + \eta), y_i),$$

where $\eta \sim \mathcal{U}(-k, k)$. The time complexity for the inner-loop of this algorithm is similar to GradAlign: $O(2,p,b)$. A visual comparison of FGSM [29], RS-FGSM (FastAT) [87] and N-FGSM is given in Figure 6.

Zhang et al. [95] revisited the FastAT formulation and reformulated it as a Bi-Level optimization problem equivalent to solving a linearized BLO problem involving a sign operator. Then, they proposed a new family of adversarial training methods called Fast Bi-Level AT.

2) ADVERSARIAL TRAINING WITH RECYCLING OF GRADIENT INFORMATION

The PGD-$k$ based adversarial training uses a $k$-step adversarial attack to accumulate attack strength and create strong adversarial examples, as shown in Equation 3. Each of the $k$ steps require computation of gradient w.r.t. input for entire batch ($\nabla_x L$). This is one reason for the success of the PGD-$k$ method. In this section, we discuss methods that devised clever ways to accumulate attack strength by recycling gradient information of an input batch across training epochs to circumvent the need for iterative computation.

The Free Adversarial Training (FreeAT) [73] is the first method to propose the reuse of gradient information. They accomplish it by “recycling gradient information”. Specifically, FreeAT creates adversarial examples with just one backward step (similar to FGSM defined in Equation 2. However, it divides the training of a model into $n/m$ replays where $n$ is the number of training epochs. Each replay consists of $m$ epochs and uses a single batch of data. Adversarial examples are created with a single back-propagation step in each training step in a replay. However, the perturbation is accumulated across the replay. This way, training accumulates adversarial examples for $m$-replays without incurring any extra cost while also being trained for $n$ epochs. The time complexity of attack for this method is also $O(p,b)$, however, the overall complexity may increase based on the number of replays used.

The Adversarial Training with Transferable Adversarial Examples (ATTA) [100] is conceptually similar to FreeAT. This method is based on a simple observation: adversarial
examples in previous epochs continue to be adversarial in subsequent epochs. ATTA stores adversarial perturbation for each input example at a certain time $x_t^{adv}$ and then reuses it in subsequent epochs.

However, naively using this method can result in sub-optimal adversarial training for two reasons. First, dynamic augmentation introduces mismatch as augmentation can be different across epochs. Second, significant changes in model parameters can make an adversarial example useless. To circumvent the first problem, ATTA introduced a connection function $C$. This connection function is comprised of an inversion of the data augmentation function ($C$) that can invert the input and can add augmentation. To solve the second problem, periodical resetting of perturbations (e.g., restart perturbation after a certain number of epochs) is used. The complexity of the attack method for ATTA is the same as adversarial training. However, the improvement comes from the reuse of adversarial perturbation across training. A visual comparison of PGD-7 and ATTA is given in Figure 7.

3) PARTIAL ADVERSARIAL TRAINING METHODS

Unlike the previous two categories, research in this direction focuses on removing redundant or less useful parts of adversarial training. In this way, the complexity of the adversarial training is reduced significantly in exchange for a small sacrifice in robustness and accuracy.

The first direction is finding parts of models that are most important for adversary update as shown in Figure 8. Zhang et al. [93] studied the effect of different layers on adversarial example update (inner-maximization loop in Equation 4) and investigated the effect of different layers on it. Through Pontryagin’s Maximum Principle (PMP), they observed that a significant part of adversarial perturbation update is only coupled with the first layer of a model. Based on this observation, they introduced You Only Propagate Once (YOPO). YOPO freezes a model except for the first layer (layer 0 to layer $l-1$ are frozen, where $l$ is the number of layers) during the back-propagation for adversary update. This makes the complexity of the YOPO significantly less compared with PGD-7 as each adversary update requires fewer backward propagation computations.

The second direction consists of utilizing a combination of normal and adversarial training as shown in Figure 9. Gupta et al. [32] investigated whether it is possible to normally train a model first and then train it on adversarial examples and proposed Delayed Delayed Adversarial Training, Delayed AT is based on the following observation: adversarial examples generated in the initial phase are quite similar to normal examples. Therefore, normal training is a good substitute for adversarial training in the starting phase. However, finding a ‘switch’ epoch (transition from normal to adversarial one) is not trivial to find as a fully-trained model can not be used. To solve this problem, Delayed AT also introduced a method to find an optimal time to switch.

The switch for DelayedAT has based on the observation that the strength of the adversary becomes reasonable at the point when model parameters start to converge on local minima. This convergence is determined by the following process: if the training loss value at the current epoch is within $D\%$ of the running average of training losses over previous $W$ epochs. In conclusion, the model is trained with natural examples while keeping track of model parameters. When model parameters start converging (based on stable loss value), adversarial training is switched on. This switch is usually turned on at a later stage, making the cost of the adversarial training significantly lower.

The third direction is about exploring adversarially training the model on a representative subset of the whole dataset as shown in Figure 10. Dolatabadi et al. [20] first investigated the effect of adversarial training with a subset of original data that sufficiently covers important attributes of the data. To find a representative subset of the data that can be useful for adversarial training, they utilized coreset selection methods [61]. However, coreset selection methods are designed for normal training. To make these methods compatible with adversarial training, they proposed to use coreset selection with the gradient of the whole dataset. Specifically, the algorithm consists of finding an approximation of the whole dataset via. gradient matching and then training the
model on this best subset. The two processes are carried out alternatively.

Hua et al. [39] also suggested training models on a small subset of the original data. However, they proposed a method to find the subset of data by mining boundary examples. Specifically, they show examples that exist close to the model boundary and have the maximum effect on the adversarial training. These examples, also called boundary examples are mined by using the model’s miss-classification and uncertainty.

Boundary examples are inputs that are classified correctly without perturbation but miss-classified after the addition of perturbation. Furthermore, the model’s uncertainty (Var[Softmax(f(x))] for these examples can be exploited to further distinguish these examples. Based on these observations, the authors introduced signed prediction variance (SVar[.]). The sign of the prediction (negative if misclassified, positive otherwise) are multiplied with variance to get the signed variance. All examples that fall in a certain top percentile of examples are classified as boundary examples. The model is then trained on these boundary examples.

B. TRANSFERABILITY OF ROBUSTNESS

In this section, we overview methods that leverage different transfer learning frameworks to transfer robustness from a pre-trained model. We divide them into two subcategories: transfer learning-based methods and distillation-based methods. A quantitative comparison of these methods is given in Table 3 and Figure 11.

1) TRANSFER LEARNING-BASED METHODS

Transfer learning, learning on one dataset and transferring this learned knowledge to another dataset, is a famous paradigm in deep learning [21], [41], [47], [91]. However, it has primarily been explored in the context of natural training. Can we extend it for robustness transfer?

Shafahi et al. [74] investigated this question and showed that a naive extension of transfer learning methods may not work as well for robustness transfer. First, they showed that a model keeps its robustness if only the classification layer is retrained. Second, more number of layers being retrained translates into lesser and lesser robustness transfer. Based on these observations, they hypothesized that a model’s robustness is due to its robust features. They also showed a connection between transferability of adversarial robustness and similarity of source and target datasets. Furthermore, they also showed that a transfer learning-based regime is more suitable in a low-data regime. Finally, they introduced a simple mechanism to fine-tune the whole network while keeping the robustness intact. They proposed to match intermediate layers of a normally trained model with a fixed robust feature extractor.

Similar to Robust Transfer Learning [74], Hendrycks et al. [34] also investigated the effect of a robust pre-training from the context of a larger dataset pre-training. They showed that it is possible to improve robustness if the model is adversarially pre-trained on a large dataset.

2) DISTILLATION OF ROBUSTNESS

Knowledge distillation is a well-explored framework in deep learning. It consists of training a student model (denoted as fθ) more efficiently with the help of a teacher model (fT). The distillation is often achieved by minimizing one of the following losses: loss between the labels of teacher and student [10], [37], reducing the distance between activations of both models on the same inputs [35], [36], [40], [45], [46], [69], [76], [80], [82], [90], [92] or by matching gradients [78], [92]. However, all of these methods are designed for knowledge transfer under normal training. Recently, a number of methods have been introduced to transfer robustness from a teacher to a student model. An illustration of the possible ways for robustness distillation is given in Figure 13. We review these methods in the following sections.

a: FROM LABELS

The distillation was originally proposed by Hinton et al. [37] to transfer the knowledge of a large pre-trained teacher to a student by making student models mimic labels produced by the teacher. Goldblum et al. [27] introduced a simple way to extend this distillation loss to transfer the robustness. They use the same formulation proposed by Hinton et al. [37] for training but investigated different experimental configurations to transfer the robustness. Specifically, they show that it is possible to transfer robustness from a larger model (e.g., ResNet) to a smaller model (e.g., MobileNet). They also showed that distillation is possible without requiring to generate adversarial examples, however, using efficient adversarial training methods (e.g., FastAT) can improve the robustness transfer significantly. Furthermore, they showed...
that not all robust models are good at transferring robustness. The loss function for robust distillation training can be written as follows.

\[
L_{RD} = (1 - \alpha)L_{CE}(f^S(x), y) + \alpha t^2 L_{RKD}(f^S(x_{adv}), f^T(x)),
\]

where \( t \) is the temperature term, and \( x_{adv} \) is an adversarial example created with fast adversarial training as defined in Equation 5, \( \alpha \) is the relative weight for each loss term, and \( L_{RKD} \) is robust knowledge distillation loss defined as follows.

\[
L_{RKD} = KLD(f^T(x)|t, f^S(x)|t),
\]

where \( KLD \) is KL Divergence. An overview of this method is shown in Figure 12.

The robust distillation [27] is limited to distilling the robustness to smaller, mobile-first models. Zi et al. [103] extended this work and showed that robust soft labels can improve the robustness of several state-of-the-art adversarial training methods, such as AT [57], TRADES [94], MART [85], etc. Then, they proposed a method called Revisiting Soft Robust Soft Label Adversarial Distillation (RSLAD). The RSLAD replaces cross entropy with KLD loss. The loss function for this method can be defined as follows.

\[
L_{RSLAD} = (1 - \alpha)L_{RKD}(f^T(x), f^S(x)) + \alpha L_{RKD}(f^S(x_{adv}), f^T(x))
\]

where \( x_{adv} = \arg \max_{x_{adv}} \{x_{adv} \sim x | \| x - x_{adv} \|_2 \leq \epsilon \} \) \( \) and temperature is set to be 1.

Maroto et al. [58] extended these two works and performed a more detailed study and showed the effect of different components on robustness transfer. They demonstrated that early stopping, label mixing, teacher robustness, an ensemble of teachers, and weak adversarial training improve robustness transfer. Furthermore, they showed that AKD reduces the confidence in easy examples while increasing the confidence in harder ones. This allows the student to learn them more efficiently.

\[ b: FROM INPUT GRADIENTS \]

Inspired from the previous observations that robust models have visibly salient gradients [25], [81], Chan et al. [14] proposed to distill robustness by matching input gradients. The value of input gradients represents how small changes can affect the model’s input. Therefore, it can represent the importance of each pixel for the output. The authors proposed to distill the importance of a robust teacher by matching its robust input gradients with the gradients of the student. This resulted in a model and data agnostic robustness distillation method.

IGAM (Input-Gradient Adversarial Matching) matches the input gradient of teacher and student via. a discriminator which follows GAN [28] like training strategy. Specifically, the student mimics the input gradients of the teacher in such a way that a trained discriminator can not distinguish between the input gradients of robust teacher and student. The loss function for the discriminator of IGAM is formulated as follows:

\[
\mathcal{L}_{Disc} = \mathbb{E}_{J_T} [\log f_{disc}(J_T)] + \mathbb{E}_{J_S} [\log (1 - f_{disc}(J_S))].
\]

where \( J = \nabla_x \mathcal{L} \) is the input gradient for teacher \( f^T \) and student model \( f^S \). An illustration of this part of IGAM is shown in Figure 14. The authors also proposed to add a gradient regularization term.

\[
\mathcal{L}_{Diff} = \| J_S - J_T \|_2^2.
\]
Finally, the overall training objective function for IGAM can be written as follows.

\[ \mathcal{L}_{IGAM} = \arg \min_{\theta} \mathcal{L}_{CE} + \gamma_{Dis} \cdot \mathcal{L}_{Dis} + \gamma_{Diff} \cdot \mathcal{L}_{Diff}, \]

where \( \gamma \) represents the weight for respective loss functions.

Maroto et al. [58] proposed to distill robustness with both input gradients and soft labels. They proposed a method called knowledge distillation with gradient alignment (KDIGA). The loss function for KDIGA is defined as follows.

\[ L_{kdiga} = \min_{\theta} \gamma_{CE} L_{CE} + \gamma_{RKD} L_{RKD} + \gamma_{IGA} L_{IGA} \]

where \( L_{IGA} \) is same \( L_{Diff} \) from IGAM (Equation 7) and \( mathcal{L}_{RKD} \) is same defined in AKD (defined in Equation 6). Their proposed method achieved good robustness transfer across architectures such as ResNets [33] and Visual Transformers [23].

c: FROM INTERNAL REPRESENTATIONS

Awais et al. [6] proposed to distill robustness via intermediate activations of the robust teacher model. They argued that adversarial examples destroy class-related information by suppressing class-related activations of a model. Adversarial training learn to avoid this issue. Hence they proposed to match the activation patterns of a robust teacher and a student model. They propose to match the activations maps by minimizing the following loss.

\[ L_{acm}(A^T_i, A^S_i) = \| \frac{A^T_i}{\| A^T_i \|_2} - \frac{A^S_i}{\| A^S_i \|_2} \|_2^2, \]

where acm stands for activated channel maps and \( a_i \) for each layer are extracted by applying a function on \( i \)-th block activations \( A_i \) (output of \( i \)-th layer), \( g_c : \mathbb{R}^{B \times C \times H \times W} \rightarrow \mathbb{R}^{B \times C \times 1 \times 1} \). The ACM loss for all layers can be defined as: \( L_{acm} = \sum_i L_{acm} \). They also proposed to add mixup augmentation and KL loss. Their overall loss is defined as,

\[ L_{MixACM} = (1 - \gamma_{kld}) L_{mixup} + \gamma_{kld} L_{kld} + \gamma_{acm} L_{acm}, \]

where \( \gamma \) is the weight for respective loss terms.

Vaishnavi et al. [84] argued to match the representations of the only penultimate layer of teacher and student. Their loss function can be written as follows.

\[ L = \gamma L_{CE}(f^S(x)) + \mathcal{L}_R(g^S(x), g^T(x)), \]

where \( g \) is the deep model without a classification layer, and \( \mathcal{L}_R \) is the loss between penultimate layers of teacher and student models.

Li et al. [54] argued that small adversarial changes in the input space got magnified and cause “attention divergence”. They proposed to keep the attention by using an adversarially trained teacher. More specifically, they defined the following loss function for attention transfer.

\[ L_{at} = \sum_{l \in L} \| A^T_l(x) - A^S_l(x_{adv}) \|_p + \| A^T_l(x) - A^S_l(x) \|_p, \]

where \( A^l \) is a function that maps the \( l \)-th layer to the attention map taken from Zhou et al. [101]. This method improves the results of baseline adversarial training on various methods.

IV. AN OUTLOOK ON LIMITATIONS OF PREVIOUS WORKS AND FUTURE DIRECTIONS

In the previous sections, we have presented a holistic review of the literature for efficient adversarial robustness methods in deep learning. While these sections provided a technical
overview of the methods along with their results, below we present a general overview of the limitations of these works along with the possible future research directions. The purpose of this section is to provide a guide for future research in this field.

A. BRIDGING THE PERFORMANCE GAP IN SINGLE-STEP ADVERSARIAL TRAINING METHODS AND ADVERSARIAL TRAINING

Single-step adversarial training methods have achieved significant improvement in terms of efficiency compared with adversarial training. However, there is still a gap in terms of robustness and accuracy.

The main challenge in single-step adversarial attacks-based training is the proper approximation of the inner loop for Equation 4 in a single step. The accurate approximation requires a theoretically motivated explanation of FastAT. Recent expositions on why FastAT works [3], [15] have shed some light on this problem, but it remains an avenue to explore. A theoretically motivated explanation of the reasons why FastAT works can not only improve the robustness but also the accuracy of fast adversarial training.

Furthermore, single-step methods can accomplish improvements in efficiency by training on smaller-size datasets. Recent works have shown the potential of the effective-sample-selection in both normal as well as adversarial training [12], [20], [39], [61], [97], [98], [99]. Utilization of single-step methods along with dataset selection methods can be an interesting future direction to improve both the efficiency and performance of robustness methods.

Finally, most single-step methods only focus on improving the efficiency of the inner loop of adversarial training. However, like FastAT [87], it is possible to use auxiliary methods to further improve the efficiency of these methods. For instance, like FastAT, it is possible to use better learning rate scheduling methods (e.g., cyclic learning rate by Smith [77]) and mixed precision arithmetic [59].

B. IN-DEPTH EXPLORATION OF PARTIAL ADVERSARIAL TRAINING

As discussed in Section III-A3, partial training methods have explored some redundant aspects of adversarial training, such as data, model, and training. However, a more systematic study can reveal better ways to improve adversarial training. We list some of the research directions here.

First, a systematic study to understand the combined relation between three aspects, the intermediate layers required to get a good and efficient adversary update (as studied by Zhang et al. [93]), the number of data samples required for the convergence of training (e.g., as studied by Dolatabadi et al. [20] Hua et al. [39]), and alteration of normal and adversarial training (as studied by Gupta et al. [32]), can help improve adversarial training significantly. This is because such a study will be able to find a sweet spot between these three crucial factors.

Second, an important aspect of efficient adversarial training is data. It has been shown that all data samples are not created equal for adversarial training [39]. Based on this observation, a few studies have taken initial steps to find a minimum set of samples that can effectively represent the complete dataset. However, its largely been explored in the context of normal training and only a few directions have been explored for adversarial robustness. For instance, one possible direction, that has not yet been explored in the context of adversarial robustness, is to get the representative subset of the dataset through data condensation [12], [97], [98], [99]. Data condensation learns to synthesize a small set of informative data samples from a large dataset. Data condensation decreases the size of the data significantly thereby making the training time extremely short. An adversarial-aware data condensation method can improve both the performance and efficiency of adversarial training methods.

C. EXPLOITATION OF TRANSFER LEARNING AND DISTILLATION METHODS

Transfer learning and distillation have been explored for adversarial robustness. However, the scope of this exploration is still limited. A significant amount of work has been done on transfer learning [21], [41], [47], [91] and the distillation of knowledge [10], [35], [36], [37], [40], [45], [46], [69], [76], [78], [80], [82], [90], [92] for normal training. Borrowing some lessons from this literature can lead to more efficient and accurate transferability of robustness. We list a few possible directions here.

First, the potential of transfer learning has only been explored in the context of normal retraining. More specifically, Shafahi et al. [73] have shown that a model loses its robustness if all of its layers are retrained on a new dataset. However, it is possible to retrain a robust model on a new
dataset with adversarial examples. This may decrease the number of samples required for retraining thereby decreasing training time significantly.

Second, the training time of the above-mentioned transfer learning-based adversarial training can further be reduced by using adversarial examples produced through an efficient method such as FastAT [87]. Furthermore, only using a small but effective subset of training samples for adversarial training can make this process significantly more efficient and can also improve performance.

Third, the knowledge distillation for the robustness transfer has also been explored in a limited context. For instance, Chan et al. [14] proposed to match input gradients of robust teacher and student via. a discriminator. However, previous work on normal distillation [78] has shown the effect of Jacobian matching on not only clean accuracy but also noise robustness. Therefore, a Jacobian-based adversarial robustness transfer method has the potential to be more effective and efficient at the same time.

Finally, the intermediate features play a critical role in knowledge distillation. A plethora of work exists that has shown the effectiveness of exploiting these features in a multitude of ways [10], [35], [36], [37], [40], [45], [46], [69], [76], [78], [80], [82], [90], [92]. This previous work can be exploited in the context of adversarial training. For instance, one work that closely relates to robustness transfer is by Kim et al. [45]. They proposed to utilize intermediate features by first passing them through a paraphrasing network. This paraphrasing network translates the knowledge into a form that is easily digestible for the student network. In robustness distillation, the teacher is trained in a way that is significantly different from the student. For this reason, a small paraphrasing network can improve the performance of a student by making the intermediate features of the teacher more aligned with the student.

D. COMBINATION OF DIFFERENT TECHNIQUES
A major limitation of efficient robustness methods is that most of them are studied in isolation. More specifically, each work studies one particular aspect of efficiency. For instance, single-step methods try to make inner-loop (form Equation 4) more efficient, partial training methods utilize redundancies in different aspects of models or datasets, and transfer learning methods employ pre-trained models. However, there is a need for a study that utilizes a combination of all of these to fully utilize their potential.

We provide a concrete example here. It is possible to use efficient single-step methods (e.g., FastAT [87] or N-FGSM [18]) to generate efficient adversarial examples on the most effective subset of data (selected with coreset-selection [20] or BulletTrain [39]) while utilizing gradient from only a few but effective layers (selected by YOPO [93]). A robust teacher can further improve this whole process by providing meaningful and robust guidance. In summary, a study to understand and combine different efficient adversarial training methods can result in more effective and efficient robustness methods.

E. ANALYSIS OF EFFICIENT ADVERSARIAL TRAINING METHODS FOR DESIRABLE PROPERTIES
Adversarial training leads to adversarial robustness as well as other desirable properties, such as better general robustness, more interpretability, and transferable features [24], [42], [71], [72], [81], [83], [102]. Since the aim of efficient adversarial training methods is to approximate adversarial training while being more computationally efficient, it is important to analyze if they also keep these desirable properties intact. However, the most efficient robustness methods reviewed in this study do not consider this important aspect of robustness. For this purpose, a more systematic study to analyze these properties of efficient adversarial training methods beyond just test-set robustness is required. Such a study can, likewise, help improve the performance of existing methods by providing insights. We provide a list of possible options here.

To start with, it has been shown that adversarially trained models have better generative capabilities [24], [72], [81], [102]. It is noteworthy that normally trained classifiers do not possess this quality [72]. Similarly, interpretation of deep models is a major research problem in deep learning [13], [38], [96]. It deals with understanding the decisions of a deep model. Recent works have shown that robust models are easier to interpret compared with normally trained models [24], [72]. For instance, Engstrom et al. [24] have shown that representations of robustly trained models align much more closely with human-interpretable features. Finally, adversarially trained models have been shown to learn better and more transferable features [71]. A comprehensive study to understand these aspects of efficient adversarial training methods is required.

V. CONCLUSION
In this paper, we surveyed efficient adversarial robustness methods for the first time. We focused on providing a holistic overview that can be advantageous for future developments. The high computational cost of adversarial training led to the development of efficient robustness methods. These methods leverage various inefficiencies in adversarial training and utilize different efficient training methods. These efficient methods include approaches that use single-step attacks, recycle gradient information across training epochs, decrease training or data complexity, and transfer and distill knowledge from robust pre-trained models. These methods have ushered significant progress in reducing the computational complexity of acquiring adversarial robustness, but there is still a gap between performance and efficiency. Most efficient robustness approaches have explored a single direction to improve efficiency. Our survey of the literature suggests a multitude of ways to enhance adversarial training efficiency. Using a systematic approach to explore all possible directions can result in better efficient adversarial robustness algorithms.
REFERENCES

[1] N. Akhtar and A. Mian, “Threat of adversarial attacks on deep learning in computer vision: A survey,” IEEE Access, vol. 6, pp. 14410–14430, 2018.

[2] M. Andriushchenko, F. Croce, N. Flammarion, and M. Hein, “Square attack: A query-efficient black-box adversarial attack via random search,” in Proc. Eur. Conf. Comput. Vis. Cham, Switzerland: Springer, 2020, pp. 484–501.

[3] M. Andriushchenko and N. Flammarion, “Understanding and improving fast adversarial training,” in Proc. Adv. Neural Inf. Process. Syst., vol. 33, 2020, pp. 16048–16059.

[4] A. Athalye, N. Carlini, and D. Wagner, “Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples,” in Proc. Int. Conf. Mach. Learn., 2018, pp. 142–158, Jan. 2016.

[5] M. Awais, F. Shamshad, and S.-H. Bae, “Towards an adversarially robust normalization approach,” 2020, arXiv:2006.11007.

[6] M. Awais, F. Zhou, C. Xie, J. Li, S.-H. Bae, and A. Li, “MixACM: Mixup-based robustness transfer via distillation of activated channel maps,” in Proc. Adv. Neural Inf. Process. Syst., vol. 34, 2021, pp. 4555–4569.

[7] M. Awais, F. Zhou, H. Xu, L. Hong, P. Luo, S.-H. Bae, and Z. Li, “Adversarial robustness for unsupervised domain adaptation,” in IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2021, pp. 8568–8577.

[8] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli, “Evasion attacks against machine learning at test time,” in Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases. Berlin, Germany: Springer, 2013, pp. 387–402.

[9] T. B. Brown, D. Mané, A. Roy, M. Abadi, and J. Gilmer, “Adversarial exemplars,” 2017, arXiv:1712.09665.

[10] C. Bucilua, R. Caruana, and A. Niculescu-Mizil, “Model compression,” in Proc. 12th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, 2006, pp. 535–541.

[11] N. Carlini and D. Wagner, “Towards evaluating the robustness of neural networks,” in Proc. IEEE Symp. Secur. Privacy (SP), May 2017, pp. 39–57.

[12] G. Cazanavette, T. Wang, A. Torralba, A. A. Efros, and J.-Y. Zhu, “Dataset distillation by matching training trajectories,” in IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2022, pp. 4750–4759.

[13] S. Chakraborty, R. Tomsett, R. Rahgavendra, D. Harborne, M. Alizantot, F. Cerutti, M. Srivastava, A. Preece, S. Julier, and R. M. Rao, “Interpretability of deep learning models: A survey of results,” in Proc. IEEE SmartWorld, Ubiquitous Intell. Comput., Adv. Trusted Comput., Scalable Comput. Commun., Cloud Big Data Comput., Internet People Smart City Innov. (SmartWorld/SCALCOM/UCIC/AITC/BCOM/IOP/JSPC), Aug. 2017, pp. 4–6.

[14] A. Chan, Y. Tai, and Y.-S. Ong, “What it thinks is important is important: Reluplex: An efficient SMT solver for verifying deep neural networks,” in Proc. 32nd Int. Conf. Neural Inf. Process. Syst., 2019, pp. 18527–18538.

[15] G. Hinton, O. Vinyals, and J. Dean, “Distilling the knowledge in a neural network,” 2015, arXiv:1503.02531.

[16] F. Hohman, M. Kahng, R. Pienta, and D. H. Chau, “Visual analytics in deep learning: An interrogative survey for the next frontiers,” IEEE Trans. Vis. Comput. Graph., vol. 25, no. 5, pp. 2674–2693, Aug. 2019.

[17] W. Hua, Y. Zhang, C. Guo, Z. Zhang, and G. E. Suh, “Bullettrain: Accel- erating robust neural network training via boundary example mining,” in Proc. Adv. Neural Inf. Process. Syst., vol. 34, 2021, pp. 18527–18538.

[18] S. Huang and N. Wang, “Like what you like: Knowledge distill via neuron activation feature,” in Proc. Adv. Neural Inf. Process. Syst., vol. 33, 2020, pp. 2674–2693, Aug. 2019.

[19] A. Ilyas, S. Santurkar, L. Engstrom, B. Tran, and A. Madry, “Adversarial training,” 2022, arXiv:2202.01181.

[20] B. Heo, J. Kim, S. Yun, H. Park, N. Kwak, and J. Y. Choi, “A comprehensive overhaul of feature distillation,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2019, pp. 1921–1930.

[21] B. Heo, M. Lee, S. Yun, and J. Y. Choi, “Knowledge transfer via distillation of activation boundaries formed by hidden neurons,” in Proc. AAAI Conf. Artif. Intell., 2019, pp. 3779–3787.

[22] H. Hwang and N. Wang, “Like what you like: Knowledge distill via neuron selectivity transfer,” 2017, arXiv:1707.01219.

[23] M. Huh, P. Agrawal, and A. A. Efros, “What makes ImageNet good for transfer learning?” 2016, arXiv:1608.08614.

[24] A. Byas, S. Santurkar, L. Engstrom, B. Tran, and A. Madry, “Adversarial examples are not bugs, they are features,” in Proc. Adv. Neural Inf. Process. Syst., vol. 32, 2019, pp. 1–15.

[25] J. Kang, S. Tariq, H. Oh, and S. S. Woo, “A survey of deep learning-based object detection methods and datasets for overhead imagery,” IEEE Access, vol. 10, pp. 20, 2018–2034, 2022.

[26] G. Katz, C. Barrett, D. L. Dill, K. Julian, and M. J. Kohrenderfer, “Reluplex: An efficient SMT solver for verifying deep neural networks,” in Proc. Int. Conf. Comput. Aided Verification. Cham, Switzerland: Springer, 2017, pp. 97–117.

[27] J. Kim, S. Park, and N. Kwak, “Paraphrasing complex network: Network compression via factor transfer,” in Proc. 32nd Int. Conf. Neural Inf. Process. Syst., 2018, pp. 2765–2774.

[28] D. Pedreschi, “A survey of results,” in Proc. 32nd Int. Conf. Neural Inf. Process. Syst., 2019, pp. 3509–3518.

[29] S. Kornblith, J. Shlens, and Q. V. Le, “Do better ImageNet models transfer better?” 2016, arXiv:1912.02768.

[30] F. Hohman, M. Kahng, R. Pienta, and D. H. Chau, “Visual analytics in deep learning: An interrogative survey for the next frontiers,” IEEE Trans. Vis. Comput. Graph., vol. 25, no. 5, pp. 2674–2693, Aug. 2019.

[31] W. Hua, Y. Zhang, C. Guo, Z. Zhang, and G. E. Suh, “Bullettrain: Accelerating robust neural network training via boundary example mining,” in Proc. Adv. Neural Inf. Process. Syst., vol. 34, 2021, pp. 18527–18538.

[32] S. Huang and N. Wang, “Like what you like: Knowledge distill via neuron selectivity transfer,” 2017, arXiv:1707.01219.

[33] M. Huh, P. Agrawal, and A. A. Efros, “What makes ImageNet good for transfer learning?” 2016, arXiv:1608.08614.

[34] A. Byas, S. Santurkar, L. Engstrom, B. Tran, and A. Madry, “Adversarial examples are not bugs, they are features,” in Proc. Adv. Neural Inf. Process. Syst., vol. 32, 2019, pp. 1–15.

[35] J. Kang, S. Tariq, H. Oh, and S. S. Woo, “A survey of deep learning-based object detection methods and datasets for overhead imagery,” IEEE Access, vol. 10, pp. 20, 2018–2034, 2022.

[36] G. Katz, C. Barrett, D. L. Dill, K. Julian, and M. J. Kohrenderfer, “Reluplex: An efficient SMT solver for verifying deep neural networks,” in Proc. Int. Conf. Comput. Aided Verification. Cham, Switzerland: Springer, 2017, pp. 97–117.

[37] J. Kim, S. Park, and N. Kwak, “Paraphrasing complex network: Network compression via factor transfer,” in Proc. 32nd Int. Conf. Neural Inf. Process. Syst., 2018, pp. 2765–2774.

[38] D. Pedreschi, “A survey of results,” in Proc. 32nd Int. Conf. Neural Inf. Process. Syst., 2019, pp. 3509–3518.

[39] S. Kornblith, J. Shlens, and Q. V. Le, “Do better ImageNet models transfer better?” 2016, arXiv:1912.02768.
A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017.

A. Krakin, J. J. Goodfellow, and S. Bengio, “Adversarial examples in the physical world,” 2016, arXiv:1607.02533.

A. Krakin, J. J. Goodfellow, and S. Bengio, “Adversarial examples in the physical world,” in *Artificial Intelligence Security and Safety*, Boca Raton, FL, USA: CRC Press, 2018, pp. 99–112.

Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” in *Proc. 25th Int. Conf. Mach. Learn.*, New York, NY, USA: ACM, 2012, pp. 1097–1104.

A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Learning multiple layers of features from tiny images,” *Adv. Neural Inf. Process. Syst.*, vol. 25, 2012, pp. 1097–1105.

M. A. Soltani, A. Ilyas, L. Engstrom, A. Kapoor, and A. Madry, “Do adversarially robust ImageNet models transfer better?” 2020, arXiv:2007.08489.

S. Santurkar, A. Ilyas, D. Tsipras, L. Engstrom, B. Tran, and A. Madry, “Image synthesis with a single (robust) classifier,” in *Adv. Neural Inf. Process. Syst.*, 2019, pp. 1262–1273.

A. Shafahi, M. Najibi, A. Ghiassi, Z. Xu, J. Dickerson, C. Studer, L. S. Davis, G. Taylor, and T. Goldstein, “Adversarial training for free!” 2019, arXiv:1904.12843.

A. Shafahi, P. Sadatpanah, C. Zhu, A. Ghiassi, C. Studer, D. Jacobs, and T. Goldstein, “Adversarially robust transfer learning,” in *Proc. Int. Conf. Learn. Represent.*, 2019, pp. 1–14.

R. Shao, J. Yi, P.-Y. Chen, and C.-J. Hsieh, “How and when adversarial robustness transfers in knowledge distillation?” 2021, arXiv:2110.12072.

Z. Shen, Z. He, and X. Xue, “MEAL: Multi-model ensemble via adversarial learning,” in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 4886–4893.

L. N. Smith, “Cyclical learning rates for training neural networks,” in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2017, pp. 464–472.

S. Srivastava and F. Fleuret, “Knowledge transfer with Jacobian matching,” in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 4723–4731.

A. S. Rakin, J. Yi, B. Gong, and D. Fan, “Defend deep neural networks against adversarial examples with quantized activation functions,” 2018, arXiv:1807.06714.

J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” in *Adv. Neural Inf. Process. Syst.*, vol. 28, 2015, pp. 1–9.

A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, “FitNets: Hints for thin deep nets,” 2016, arXiv:1412.6550.

A. Ross and F. Doshi-Velez, “Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients,” in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 1–10.

H. Salman, A. Ilyas, L. Engstrom, A. Kapoor, and A. Madry, “Do adversarially robust ImageNet models transfer better?” 2020, arXiv:2007.08489.

S. Santurkar, A. Ilyas, D. Tsipras, L. Engstrom, B. Tran, and A. Madry, “Image synthesis with a single (robust) classifier,” in *Adv. Neural Inf. Process. Syst.*, 2019, pp. 1262–1273.

H. Zhang, Z. Zhu, and T. Goldstein, “Adversarial training for free!” 2019, arXiv:1904.12843.
[101] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, “Learning deep features for discriminative localization,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 2921–2929.
[102] Y. Zhu, J. Ma, J. Sun, Z. Chen, R. Jiang, Y. Chen, and Z. Li, “Towards understanding the generative capability of adversarially robust classifiers,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2021, pp. 7728–7737.
[103] B. Zi, S. Zhao, X. Ma, and Y.-G. Jiang, “Revisiting adversarial robustness distillation: Robust soft labels make Student better,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2021, pp. 16443–16452.

A. Muhammad, S.-H. Bae: Survey on Efficient Methods for Adversarial Robustness

AWAIS MUHAMMAD received the B.S. degree from the University of Engineering and Technology (UET) Lahore, Lahore, in 2016, and the M.S. degree from Information Technology University (ITU), Lahore, in 2019. He is currently pursuing the Ph.D. degree with the Machine Learning and Visual Computing (MLVC) Laboratory, Kyung Hee University, South Korea. From 2016 to 2019, he was a Research Associate with the Signal Processing and Information Decoding (SPIDER) Laboratory, ITU. From 2020 to 2022, he was an Assistant Researcher at the Noah’s Ark Laboratory, AI Theory Group, Huawei, Hong Kong. He is currently a Research Assistant at Sony AI, Japan. His current research interests include deep learning model architectures, adversarial robustness in the presence of distribution shifts, and transfer learning.

SUNG-HO BAE (Member, IEEE) received the B.S. degree from Kyung Hee University, Republic of Korea, in 2011, and the M.S. and Ph.D. degrees from the Korea Advanced Institute of Science and Technology (KAIST), Republic of Korea, in 2012 and 2016, respectively. From 2016 to 2017, he was a Postdoctoral Associate with the Computer Science and Artificial Intelligence Laboratory (CSAIL), Massachusetts Institute of Technology (MIT), MA, USA. Since 2017, he has been an Assistant Professor with the Department of Computer Science and Engineering (CSE), Kyung Hee University. His current research interests include model compression, interpretation, adversarial attack/defense, and neural architecture search for deep neural networks.

***