Opportunistic Use of Crowdsourced Workers for Online Relabeling of Potential Adversarial Examples

Shawqi Al-Maliki*, Faissal El Bouanani[†], Senior Member, IEEE, Kashif Ahmad*, Mohamed Abdallah*, Senior Member, IEEE, Dinh Thai Hoang[‡], Member, IEEE, Dusit Niyato[§], Fellow, IEEE, Ala Al-Fuqaha*, Senior Member, IEEE

* Information and Computing Technology (ICT) Division, College of Science and Engineering, Hamad Bin Khalifa University, Doha 34110, Qatar

[†] College of Engineering, Mohammed V University in Rabat, Morocco

[‡] School of Electrical and Data Engineering, University of Technology Sydney, Australia

[§] School of Computer Science and Engineering, Nanyang Technological University, Singapore

Abstract—Deep Neural Networks (DNNs) have achieved tremendous success in handling various Machine Learning (ML) tasks, such as speech recognition, Natural Language Processing, and image classification. However, they have shown vulnerability to well-designed inputs called adversarial examples. Researchers in industry and academia have proposed many adversarial example defense techniques. However, none can provide complete robustness. The cutting-edge defense techniques offer partial reliability. Thus, complementing them with another layer of protection is a must, especially for mission-critical applications. This paper proposes a novel Online Selection and Relabeling Algorithm (OSRA) that opportunistically utilizes a limited number of crowdsourced workers (budget-constraint crowdsourcing) to maximize the ML system's robustness. OSRA strives to use crowdsourced workers effectively by selecting the most suspicious inputs (the potential adversarial examples) and moving them to the crowdsourced workers to be validated and corrected (reabeled). As a result, the impact of adversarial examples gets reduced, and accordingly, the ML system becomes more robust. We also proposed a heuristic threshold selection method that contributes to enhancing the prediction system's reliability. We empirically validated our proposed algorithm and found that it can efficiently and optimally utilize the allocated budget for crowdsourcing. It is also effectively integrated with a state-of-the-art black-box (transfer-based) defense technique, resulting in a more robust system. Simulation results show that OSRA can outperform a random selection algorithm by 60% and achieve comparable performance to an optimal offline selection benchmark. They also show that OSRA's performance has a positive correlation with system robustness.

Index Terms—Adversarial Machine Learning, Evasion Attacks, Adversarial Examples, Adversarial Defense, Crowdsourcing, Security, Online Relabeling.

I. INTRODUCTION

DEEP Neural Networks (DNNs) have shown noticeable success in handling Computer Vision (CV) problems, such as image classification [1], object detection [2], and face recognition [3]. They also have demonstrated great success in other complicated Machine Learning (ML) tasks such as speech recognition [4] and Natural Language Processing (NLP) [5], [6]. However, DNNs have not yet shown remark-

able improvement in learning intrinsic concepts that lead to correct output labels [7]. That is, in the image classification task, as an example, images and the corresponding labels are meaningless from the DNNs perspective. There is a semantic gap (misalignment) between how computers and humans interpret the image representation. For that, they suffer from a phenomenon called adversarial examples [8]. Szegedy et al. [9] observed the presence of adversarial examples (evasion attacks) in the image classification task where it is possible to change the predicted label of the image by adding a well-designed small amount of perturbation.

In attempting to make DNNs robust against adversarial examples, various solutions have been proposed in the literature [10]–[17]. However, these countermeasures techniques have managed to only partially strengthen the DNN models. They make the DNN models more effective but not completely robust [18]. In addition, defending DNN models against adversarial examples is considered hard because the process of constructing the adversarial examples is not theoretically-backed; rather, it is an optimization problem that is non-linear and non-convex [19], [20]. For that, DNNs are not totally reliable, which restricts them from being used in safety and mission-critical applications. To augment their robustness, they need to be supported by another layer of defense.

In this work, we propose a two-stage solution where we utilize a state-of-the-art defense technique as a primary stage (Stage 1) and then complement it with an Online Selection and Relabeling Algorithm (OSRA) as an added validation layer (Stage 2). Specifically, we first form the proposed framework in two stages to decouple suspicious and non-suspicion elements making the system more efficient. That is, only the suspicious elements are affected by the additional overhead of OSRA in Stage 2. OSRA is a sliding window-based algorithm that opportunistically utilizes the crowdsourced workers to maximize the ML system robustness (i.e., reducing the number of adversarial examples by capturing and correcting the most suspicious ones). It strives to use crowdsourced workers efficiently by selecting the most suspicious elements in Stage

1 (the potential adversarial examples) and moving them to the crowdsourced workers to get validated and relabeled. In an online selection algorithm, the inputs come in a random order, and the selection is made on the spot (take it or leave it instantly), and, most importantly, we cannot revoke that selection decision [21].

Crowdsourcing is a costly process (compared with the automatic prediction process that does not require human intervention). Hence, we consider budgeted crowdsourcing, where a predefined budget is allocated to the process. An opportunistic use of crowdsourced workers indicates a strive in utilizing the allocated budget efficiently. Thus, in this paper, “Opportunistic use of crowdsourced workers” and “utilizing allocated budget efficiently” imply one another. In this paper, we assume that one allocated budget unit is used for requesting a crowdsourced worker to validate one element. So, “the allocated budget” and “the crowdsourced elements” are used interchangeably in this work. The OSRA’s ultimate goal is to minimize the allocated budget while maximizing the success rate of relabeling. This is achieved by validating the most suspicious elements following our system model as detailed in Section III.

We also consider utilizing crowdsourcing within our proposed algorithm because it works naturally well in handling image classification tasks [22]. In other words, a crowdsourced worker can easily validate a suspected image by comparing its predicted label (assigned in Stage 1) with the actual representation of that image. Section III-B2 is a detailed illustrative example that shows how our proposed algorithm works. To further enhance the proposed algorithm OSRA, we propose a heuristic threshold selection method to filter the stream of output confidences from Stage 1.

OSRA could be used in critical-mission applications, which deserves the added cost of the validation layer. A suggested mission-critical application is the medical domain (e.g., x-ray diagnosis). As detailed in Section V-C, OSRA can also be used in other applications such as face recognition, online learning, online active learning, and as also human-intervention-agnostic labeling technique.

Our hypothesis is to opportunistically use a limited number of crowdsourced workers to enhance the robustness of state-of-the-art adversarial ML defense against black-box adversarial attacks. Our proposed algorithm OSRA uses an online selection method that prioritizes the selection of the most suspicious elements to be moved to a budget-constraint crowdsourcing process to get verified and relabeled. We expect that OSRA can give a better success rate of relabeling than that of the random selection method (as the worst case baseline) and achieves a competitive performance to an optimal offline selection algorithm (as the best case benchmark). We also demonstrate theoretical and empirical proof to show that our hypothesis holds. To the best of our knowledge, this is the first work that considers budgeted crowdsourcing as a complementary layer to adversarial ML defenses.

The main contributions of the work can be summarised as follows:

- We propose an Online Selection and Relabeling Algorithm (OSRA) on top of adversarial examples defense

method to further enhance the robustness of DNNs.

- Mathematically, we prove that the proposed scheme maximizes the model’s robustness by keeping the allocated budget (the number of crowdsourced workers) lower.
- We propose a heuristic threshold selection method to filter the stream of output confidences that come from the primary stage, and hence enhance the performance of OSRA.
- To validate the effectiveness of the proposed defense scheme, we conducted extensive experiments on an image classification task under the evasion attacks, more specifically the black-box (transfer-based) threat model attacks. Through extensive experiments, we aim to answer the following research questions.
 - 1) How is the performance (the success rate of relabeling) of the proposed OSRA comparing with two baseline algorithms (the optimal offline selection algorithm and the random selection algorithm)?
 - 2) How optimal is the OSRA’s suggested sliding window size?
 - 3) How is the impact of the size of the stream of inputs, and the allocated budget on the performance of the proposed algorithm?
 - 4) How is the impact of adding the proposed heuristic threshold selection method for filtering the stream of inputs on the OSRA’s performance?
 - 5) How leveraging OSRA on top of a state-of-the-art enhances the robustness of DNNs?

The answers of these questions are detailed in Section V-E and summarized in Section V-F. The rest of the paper is organized as follows. Section II presents an overview of the related work. Section III provides the details of the proposed scheme. Section IV describes the proposed mathematical model of the proposed budgeted crowdsourcing defense scheme against adversarial attacks. Section V details the experimental set up, experimental results, discussion, and lessons learned. Finally, Section VI concludes the paper with some potential future research directions.

II. RELATED WORK

Since the introduction of adversarial examples [9], there have been three lines of research works attempting to make DNN models robust against adversarial examples. The first line of works tries heuristically (best-effort) to develop robust neural networks. The second line of works tries to achieve robustness through certified (theory-backed) approaches. The third line of works is termed as stateful detection of adversarial examples, which differs from the earlier schemes as it keeps the history of the preceding queries.

A. Heuristic Adversarial ML Defenses

In the best-effort defenses, researchers have investigated various heuristics to improve the robustness of DNNs. Examples of these heuristic techniques include DNNs distillation [13], DNNs’ inputs transformations [23], [24], generative models [11], [25], [26], adversarial training [27], ensemble training [16], [28], [29], and randomization [30], [31]. However, most

of these proposed defensive techniques have been circumvented soon after they got published [18], [32], [33]. They are not theory-backed, so the game between adversaries and defenders keeps going and the adversary wins at the end. The most effective adversarial defenses in the heuristic category are adversarial training and ensemble training methods [16]. They have withstood against adversarial attacks and achieved partial robustness. Similarly, the adversarial defenses that rely on detection (they capture the potential adversarial examples and reject them) such as auto-detection of adversarial examples [10], [34], [35], are not completely robust detectors. Tramer *et al.* [36] demonstrated that any claimed robust detector can be converted to a robust classifier, which means that detecting adversarial examples is as difficult as classifying them because adversarial defense classifiers are not yet totally robust.

Adversarial training [27] has acquired significant attention from the adversarial ML research community for its reliability and effectiveness. It is the process of crafting adversarial examples while simultaneously training DNN models with these adversarial examples after assigning them correct labels. The literature demonstrates that adversarial training urges a model to obtain robust features within datasets [8]. This process is hard and expensive, so it substantially influences the accuracy of the model on benign data [37], [38]. Alternatively, ensemble training methods have been investigated.

Ensemble approaches are considered to be the state-of-the-art technique for various ML problems [39]. They enhance the predictive power of a model by training several sub-models and merging their predicted scores. Based on the success of ensemble methods, several efforts have been made to explore the possibility of enhancing the robustness of an ensemble of sub-models against adversarial attacks, specifically transfer-based black-box attacks. The robust ensembles can be obtained by removing the shared adversarial vulnerabilities in different sub-models of the ensemble. For that, within the ensemble, the adversarial examples cannot transfer from a sub-model to another. Several works try to encourage sub-models diversity to mitigate adversarial example transferability. For instance, Pang *et al.* [28] introduce the Adaptive Diversity Promoting (ADP) regularizer, which promotes various sub-models to gain high diversity in the non-maximal predictions. Kariyappa *et al.* [29] minimize the vulnerability overlapping between various sub-models by maximizing the cosine distance between gradients in individual sub-model with respect to the input. Yang *et al.* [16] propose a vulnerability diversity metric to be utilized during the ensemble training process to ensure that the sub-models have a diverse vulnerability via minimizing the overlapped vulnerabilities within the ensemble. That is accomplished in a way that aligns well with diversifying the adversarial vulnerability shared by various sub-models. Hence, this work achieves better robustness against the transferability of adversarial attacks between sub-models. This approach is considered as the state-of-the-art in the category of ensemble training methods.

B. Certified Adversarial ML Defenses

To end the arms race between attackers and defenders related to the best-effort defensive techniques, certified ro-

bustness has emerged as a new research direction. It is in its infancy, though. Certified defenses provide a quantifiable guarantee that describes the space of inputs that produce errors. Huang *et al.* [40] develop the first certified robustness system for showing that the output label is constant throughout a specific area. Certified defenses assume that adversaries craft attacks do not exceed defined distance metrics that quantify the similarity between the original and adversarial examples. L_p norms (L_0, L_2, L_∞) are the common distance metrics used in literature to quantify similarity [18]. That is, how adversarial examples are different than the original ones. This system suffers from lacking scalability. Scaling to larger DNNs needs more strict assumptions (e.g., the prediction of a specific input point relates to a subgroup of DNN units). These kinds of assumptions imply the system can no longer offer guaranteed robustness. That is, an adversarial example that breaches the assumptions could not be detected. Reluplex [12] is another certified robustness system that utilizes linear programming solvers to size to much larger networks.

The scope of these certified robustness systems is limited because they only ensure that the perturbed inputs are predicted correctly when the perturbation results in adversarial examples that are located in a restricted area around the original input. If the perturbation results in adversarial examples are positioned far from a specific area surrounding the original input, they would not be predicted correctly. It is unattainable for a defender to fully expect all potential attacks when specifying the area encompassing x .

C. Stateful detection of adversarial examples

Under the implicit assumption that defense is a stateless function that works on the level of individual adversarial examples, making DNNs robust against adversarial inputs has demonstrated to be an extremely hard task. Most of the adversarial defenses have been shown to have significant defects [32], [41], and even those defenses that have resisted validation (such as adversarial training and ensemble training) provide only partial adversarial robustness [27].

For that, a recent promising research direction explores the potential of stateful defenses. Stateful adversarial defense differs from the earlier work defenses as it keeps the history of the preceding queries. Protecting Against DNNs Model Stealing (PRADA) [42] is an example of stateful detection of adversarial examples. Utilizing the history of queries, PRADA detects black-box model extraction adversarial attacks. It investigates only the distribution of distances. For that its scheme is not robust to the insertion of dummy queries to make the distribution Gaussian. Also, it does not consider how to capture the generation of adversarial examples.

Carlini *et al.* [17] propose a stateful technique that detects adversarial examples by maintaining a history of the previous queries, so a defender can recognize when a series of queries are suspected and could be used for creating an adversarial example. It detects query-based black-box attacks, but cannot detect transfer-based black-box attacks. Based on [17], this stateful detection strategy can be applied on top of adversarial training defenses. However, for the sake of simplicity, it

is implemented on top of a traditional (non-robust) model. Studying adversarial examples from the perspective of stateful systems is more realistic. Furthermore, it puts the defenders in a better position when responding to the adversaries [17]. However, stateful defenses do not support black-box (transfer-based) attacks and can only be applied in specific situations, where the end-user (potential adversary) is forced to create an account on the system that hosts the model, so her/his queries are tracked and audited.

As shown above, the best-effort, certified, and stateful defenses have their limitations, and they are not effective in offering complete robustness to DNN models. Enhancing DNNs' robustness through integrating the most effective and state-of-the-art best-effort defense with an extra layer of security is a promising approach to improve the robustness. This security layer could be a budgeted crowdsourcing technique (human intervention layer) or another automatic layer (e.g., another ML model). While adversarial examples can fool model-based (automatic) adversarial defenses, they cannot fool humans as adversarial examples attacks do not change the actual representation of the image. For that, our crowdsourcing-based (manual) approach can effectively capture and relabel adversarial examples. Table I categorizes the explored adversarial ML defenses based on the adversarial defense approach, the threat model, and the availability of a complementary defense layer. To the best of our knowledge, leveraging budgeted crowdsourcing as an additional validation layer on top of heuristic defensive technique is lacking. With the absence of automatic and fully robust adversarial defenses, and taking into consideration that adversarial examples fool computer vision but not human vision, a reliable human-based defensive layer is an important approach that should be adopted.

III. THE PROPOSED SYSTEM MODEL

Fig. 1 provides the block diagram of the proposed budgeted crowdsourcing defense scheme against adversarial attacks. The proposed system is composed of two main components. The first component is based on a state-of-the-art adversarial defense algorithm, while the second component is composed of the proposed budgeted crowdsourcing defense layer (OSRA) to complement the state-of-the-art defense algorithm. OSRA suggests a window size that slides over the stream of output confidence coming from the primary stage (Stage 1) and selects the minimum predicted score (potential adversarial example) within each sliding window. Then, OSRA moves the potential adversarial examples to a crowdsourcing process (Stage 2) to be validated and corrected. In the next subsections, we provide details of each component.

A. Adversarial Defense Algorithm

For the first part (Stage 1), we consider choosing a state-of-the-art adversarial defense technique that effectively mitigates the adversarial examples and minimizes their transferability. An adversarial defense algorithm (Ensemble of DNNs, as an example of the state-of-the-art defense techniques) is used as a

primary layer that induces predictions with relatively reliable confidences (correlated with the correct labels). In other words, the relationship between the correct labels and the probability of confidence of the predicted labels are maintained. The higher output confidence is, the higher probability that the predicted label is correct. Hence, the model's output (i.e., confidence scores) will have a meaningful and valid order if they are arranged in ascending or descending order. The relative reliability in output confidences that comes from partially robust models is better than the traditional models [27] where models are not incorporated with adversarial defense techniques. Thus, if an adversarial image is fed into a traditional model, it will be misclassified with high output confidence of that wrong label (i.e., the correlation is not maintained).

On the journey of finding the most effective defense technique, we investigated several state-of-the-art adversarial defense techniques [16] [28] [29] and found that the algorithm proposed in [16] is one among the best algorithms for our proposed framework as it has the best reliable outputs and can maintain high accuracy on the clean data. Moreover, it is the most effective defense against black-box (transfer-based) attacks [16].

This algorithm is based on a vulnerability diversity metric that is utilized during the ensemble training process to ensure that the ensemble's sub-models have a diverse vulnerability. The diverse vulnerability was accomplished through minimizing the overlapped vulnerabilities in the sub-models, which was done in a way that aligns well with diversifying the adversarial vulnerabilities of the ensemble's sub-models. For that, this work achieves better robustness against the transferability of adversarial attacks between sub-models. This approach can be considered as the state-of-the-art in the category of ensemble training methods.

The experiments in [16] show a correlation between the output confidence and the correct prediction. It is a good example of work that aligns well with our proposed technique. Thus, we run our proposed complementary defense technique on top of this work.

B. The Budgeted Crowdsourcing: A Complementary Validation Layer

In the proposed budgeted crowdsourcing layer (Stage 2), the confidence of each element forwarded by the previous layer (i.e., state-of-the-art adversarial defense algorithm) is compared against a set of previous elements depending on the sliding window size. The label of an element assigned by the state-of-the-art algorithm is retained if its confidence is higher than all the values in the window. Otherwise, the element is forwarded to the crowdsourcing process, where humans are asked to analyze the element and retain or change its label. In the next subsections, we provide a pseudo code of the proposed OSRA algorithm and explain it with a numerical example.

1) *The Proposed Algorithm OSRA*: In a budget-constraint crowdsourcing situation where we are restricted to a limited number of crowdsourced workers, our proposed algorithm (OSRA) opportunistically utilizes the crowdsourced workers

TABLE I: Comparisons of Different Defense Techniques Against Evasion Attacks

Reference	Adversarial Defense Approach			Threat Model			Complementary Validation Layer	
	Best-Effort (Heuristic)	Certified Robustness	Stateful Defense	White-Box	Black-Box (Transfer-Based)	Black-Box (Query-Based)	Model-Based (Auto)	Crowdsourcing-Based (Manual)
[13], [23], [25], [30], [34], [36]	✓			✓				
[27], [8], [16]	✓				✓			
[15], [43], [44]		✓						
[17], [42]			✓			✓	✓	
This work	✓				✓			✓

Fig. 1: Block diagram of the proposed budgeted defense scheme against evasion attacks.

to maximize the ML system robustness. OSRA strives to use crowdsourced workers efficiently. It selects the stream's prediction scores that have minimal prediction scores and moves them to the crowdsourced workers to be validated and relabeled. In other words, the elements with the corresponding selected prediction scores are the potential adversarial inputs that OSRA affords to validate and relabel. By this, the number of adversarial inputs gets minimized, and accordingly, the ML system becomes more robust.

OSRA utilizes the output confidences streaming from the ensemble of models (Stage 1). It takes the stream of output confidences and does the following: 1) It suggests the optimal sliding window size based on the given allocated budget and the number of expected inputs to be predicted. 2) It compares each output confidence with the previous latest inputs found in the optimal sliding window size. 3) If the confidence is found suspicious, OSRA moves it to a budgeted crowdsourcing process to get validated and corrected (relabelled). Otherwise, OSRA accepts the prediction scores that come from Stage 1 and sends them directly to the final output (without crowdsourcing).

The inputs of OSRA are: X (the elements - output confidences - coming from Stage 1), N (the number of the elements), b (the allocated budget), a sliding window of size w (proposed by OSRA based on N and b). Also, *acomparisonflag* (set to *zero* as a default value) is used to help in deciding whether the upcoming element is picked up (to go through a comparison process) or skipped (considered as a final output). The comparison process decides whether or not to move the selected element to the crowdsourced worker.

Upon the arrival of an output confidence (an element $X[i]$) to Stage 2, OSRA checks its satisfaction to two conditions: the availability of the budget (b), and the eligibility of the element to be picked up for a comparison process (i.e., a comparison flag is set to 1, which is the default value). If the former condition is not satisfied (which is the default setting), a set of output confidences of size w are moved to the final output. These elements are also copied into a buffer containing a list of comparison windows. Then, the comparison flag is set to 1. In case the latter condition is not satisfied, OSRA stops working, and the remaining elements in the stream are moved to the final output. However, if both conditions are met ($ComparisonFlag = 1$ and $Budget > 0$), the element under investigation (the picked up element) is moved to the comparison process to be checked against the confidences in the previous window (the proposed window of size w). If the selected output confidence is found less than all the output confidences in that window, OSRA moves it to the crowdsourcing process, deactivates the comparison flag (i.e., sets it to *zero*), and shifts the comparison window w steps forward. A crowdsourced worker validates the corresponding class for the selected element and either relabels it (if Stage 1 has identified it as a wrong class) or confirms that the initially assigned class is correct. OSRA converts the output confidence for the validated element to 1 as it is now validated and confirmed, and then moves it as a final output and decreases 1 from the *AllocatedBudget*. On the other hand, if the picked up element is found bigger than one of the output confidences in the previous window, OSRA moves it as a final output, and the comparison window shifts only one step forward. To better understand how OSRA works, we provide an illustrative example (Section III-B2) that demonstrates the life cycle for a sample stream of output confidences.

2) *Numeric Illustration of the Algorithm:* This numeric example simulates the role of OSRA in handling an image classification task.

As soon as the first element in the stream (with output confidence 1.0) reaches Stage 2, OSRA checks out the status

Algorithm 1 The Proposed Algorithm OSRA

Input: N : the number of elements to be investigated X : the elements to be investigated b : the allocated budget**Output:** w : the suggested sliding window size (based on N and b)

List of the selected elements to be crowdsourced

```

1:  $ComparisonFlag \leftarrow 0$ 
2:  $i \leftarrow 0$ 
3: while  $i \leq N$  do
4:   if  $ComparisonFlag = 0$  or  $AllocatedBudget \leq 0$  then
5:     if  $ComparisonFlag = 0$  then
6:       A set of elements of size  $w$  are moved as final outputs
7:       They are copied to a buffer containing a list of comparison windows
8:        $ComparisonFlag = 1$ 
9:     else if  $AllocatedBudget = 0$  then
10:      OSRA stops working
11:      The remaining elements are moved as final outputs
12:    end if
13:  else
14:     $X[i]$  is picked up and moved to the comparison process
15:     $MIN \leftarrow \text{Minimum}(\text{Comparison Window } (X[i - w : i - 1]))$ 
16:    if  $X[i] \leq MIN$  then
17:       $X[i]$  is moved to crowdsourcing process
18:      The comparison window is shifted forward by  $w$  steps
19:       $ComparisonFlag \leftarrow 0$ 
20:      Crowdsourcing worker validates  $X[i]$ 
21:      if  $X[i] \neq \text{TheGroundTruth}$  then
22:         $X[i]$  is relabeled (assigned the correct class)
23:      else
24:         $X[i]$  is validated (confirmed that the initially assigned class was correct)
25:      end if
26:      The output confidence of  $X[i]$  becomes 1.0
27:       $X[i]$  is moved as a final output
28:      The  $AllocatedBudget$  is decreased by 1
29:    else if  $X[i] > MIN$  then
30:       $X[i]$  is moved as final output
31:       $i \leftarrow i + 1$  (The comparison window is shifted forward by one step)
32:    end if
33:  end if
34: end while

```

of *ComparisonFlag* and the *AllocatedBudget*. It finds that *ComparisonFlag* = 0 (i.e., this condition is not satisfied). Thus, OSRA moves three elements (corresponding to the size of the proposed sliding window) as final outputs (without going through the crowdsourcing process), copies them to the buffer, and sets *ComparisonFlag* = 1 (Algorithm 1, lines 6-8). That creates the first comparison window (the first row in the buffer) of three output confidences, i.e., (1.0)), as well as the subsequent two output confidences (0.9, 0.6).

When the fourth element (0.3) comes to Stage 2, OSRA checks the two conditions. However, now it finds that both conditions are satisfied (*ComparisonFlag* = 1 and *AllocatedBudget* $\neq$ 0), so (0.3) is picked up and compared with the first comparison window in the buffer (1.0, 0.9, 0.6). OSRA checks the element (0.3) and finds it less than all elements in the comparison window. For that, OSRA selects and moves (0.3) to the crowdsourcing process, deactivates the *ComparisonFlag* (set it to zero), and decreases one from the *AllocatedBudget* (Algorithm 1, lines 17-19). A crowdsourced worker validates the element and notices that the initially assigned label mismatches the actual representation of the element (how the image looks), so he/she relabels it (lines 20-24). OSRA then updates the confidence to 1 and moves it as final output (lines 30-32).

Next, the fifth element in the stream (0.7) arrives at Stage 2, and since the *ComparisonFlag* is zero, OSRA moves it and two subsequent elements to final outputs, copies them to the buffer creating (0.7, 0.5, 1.0) as the second comparison window, and sets *ComparisonFlag* to 1. The eighth element in the stream (0.2) reaches Stage 2, and since *ComparisonFlag* = 1, OSRA picks it up and compares it with the latest comparison window (0.7, 0.5, 1.0). OSRA finds that (0.2) is less than all the elements in the comparison window, so it sends it to the crowdsourcing process, deactivates the *ComparisonFlag* (set it to zero), and decreases one from the *AllocatedBudget*. A crowdsourced worker validates the element (0.2) and finds that the initially assigned label mismatches the actual representation of the element, so she assigns it a new label. OSRA then updates the confidence to 1 and moves it as a final output.

After that, the ninth element (0.3) arrives at Stage 2, and since the *ComparisonFlag* is zero, OSRA moves it and the two subsequent elements to final outputs (without going through the crowdsourcing process), copies them to the buffer creating (0.3, 0.4, 0.5) as the third comparison window, and sets *ComparisonFlag* to 1.

Then, the twelfth element (0.8) in the stream reaches Stage 2, and since *ComparisonFlag* = 1, OSRA picks it up and compares it with the latest comparison window (0.3, 0.4, 0.5). However, this time OSRA observes that the picked-up element (0.8) is larger than one of the elements in the window. Thus, it is not selected, and rather, it is moved directly as the final output, and *ComparisonFlag* is set to 1 (lines 34 and 35).

A similar process continues until *AllocatedBudget* becomes zero, and then the remaining elements in the stream are moved as final outputs.

In summary, as illustrated in Fig. 2, at the end of the process of applying OSRA (presented in Section III-B1) to a stream

of twenty elements, fifteen elements were skipped: nine of them as a result of shifted windows (the *ComparisonFlag* was set to 0), while the other six elements were skipped because the *AllocatedBudget* was finished. Five elements were picked up and investigated in the comparison process, and accordingly, three of them (0.3, 0.2, 0.4) were selected (crowdsourced), and two (0.6, 0.8) were ignored. Two out of the three crowdsourced elements (0.3, 0.2) were validated and relabeled, and one (0.4) got validated but not relabeled (the initially assigned label was found correct). Accordingly, based on the performance metric (the number of corrected labels out of the crowdsourced ones), the performance of our algorithm is 66.7%. The details of the performance metric are presented in Section V.

3) *Enhancing the Proposed Algorithm with Heuristic Threshold Selection Method*: In the stream of the output confidences coming from Stage 1, we noticed that the higher the prediction score, the more reliable the prediction is. For that, we propose a heuristic threshold selection method to be applied to the stream of the prediction scores coming from Stage 1. Hence, the space of the elements being investigated becomes restricted to the most suspicious elements (the ones with lower prediction scores). To come up with the best threshold, we implemented the following reward-penalize policy (assuming that the distribution of the prediction scores of the test dataset in Stage 1 and the distributions of the stream of elements in the real-world data are the same):

- 1) We explore choosing all model prediction scores located within a relaxed uncertainty interval (the most suspect interval that contains wrong predictions) and consider them candidate thresholds. For example, considering (0.3 – 0.8) as a relaxed uncertainty interval is justified by the high confidence that the predictions with scores above 0.8 or below 0.3 are correct.
- 2) For each threshold candidate, we check only the model prediction scores below this threshold (how many were predicted correctly? how many were predicted wrongly?) Then, we assign a weight to each prediction type (1 for True prediction and -1 for False prediction) and sum up them all, getting a value.
- 3) By iterating over all threshold candidates (step 2), we come up with a vector of values (each corresponding to a candidate threshold). We select the threshold that corresponds to the largest value in that vector.

IV. MATHEMATICAL MODELING

This section proves (mathematically) that our suggested algorithm (OSRA) can efficiently select the suspicious elements (potential adversarial examples) from a stream of output confidences that come in random order. Then, OSRA moves them to reliable crowdsourced workers (assumed to be honest and highly skilled) to validate and/or correct them. To this end, the number of adversarial examples is reduced proportionally to the allocated budget, and consequently, the system becomes more robust. OSRA solves an optimization problem formulated as follows. Let N denote the number of elements to be predicted, and b denote the allocated budget

TABLE II: Summary of Mathematical Notations

Notation	Definition
N	The number of elements in the stream
X_ℓ	The carried out information, which is the output confidences corresponding to the elements in the stream.
b	The allocated budget (a positive integer) for validating the potential adversarial examples among $(X_\ell)_{1 \leq \ell \leq N}$
w	The width of the suggested sliding window, which contains a set of output confidences coming from Stage 1.
SR	Success rate of relabeling (budget utilization). It is the number of corrected (relabelled) elements out of the overall crowdsourced elements. The terms “budget utilization” and “success rate of relabeling” are used interchangeably.
Z_k	The minimum in preceding random variables of size w (counted from index $k - 1$).
A_k	The event of finding a just arrived random variable X_k greater than Z_k .
p	A positive integer such that $w + p$ is the index of the first random variable that is smaller than its w predecessors
$\mathcal{E}_{w,b}^{(p,N)}$	The event “the number of elements among $(X_\ell)_{p+w \leq \ell \leq N}$ less than their w predecessors is exactly equal to b ”

for validating and relabeling the selected elements (the most suspicious ones). Let C be the classification model. In a real-time setting where the output confidences corresponding to N elements come sequentially and arbitrarily to OSRA, and they are selected on a sliding window-based approach. At the same time, OSRA cannot revoke the selection. Find the optimal sliding window size w where b is minimum and success rate of relabeling SR is maximum. Table II summarizes the main notations used in this paper.

A. Notations

- The stream of elements coming from Stage 1 contains N information, $(X_\ell)_{1 \leq \ell \leq N}$ [16]. This information is the output confidences corresponding to the elements in the stream. As these output confidences are coming in real-time, they are considered independently and uniformly distributed Random Variables (RVs) in $[0,1]$. No knowledge is provided on the positions of the lowest output confidences, which come on the spot and in random order.
- Let w be the sliding window size, and b (a positive integer) denotes the allocated budget for validating the potential adversarial examples among $(X_\ell)_{1 \leq \ell \leq N}$
- Let Z_k refer to the minimum in preceding random variables of size w (counted from index $k - 1$).

$$Z_k = \min_{k-w \leq i \leq k-1} (X_i). \quad (1)$$

- Let $\mathcal{A}_k$ denote the event of finding a just arrived random variable X_k greater than Z_k .

$$\mathcal{A}_k = \{X_k > Z_k\}, w + 1 \leq k \leq N. \quad (2)$$

- Let p be a positive integer such that $w + p$ is the index of the first random variable that is smaller than its w predecessors, and $\mathcal{E}_{w,b}^{(p,N)}$ be the event “the number of elements among $(X_\ell)_{p+w \leq \ell \leq N}$ lesser than their w predecessors is exactly equal to b , i.e., it exists **exactly** b occurred events $\mathcal{A}_{k_1}, \mathcal{A}_{k_2}, \dots, \mathcal{A}_{k_b}$, where $\mathcal{A}_{k_i}$ denotes the complementary event of $\mathcal{A}_{k_i}$ such that

$$k_1 \geq w + p, \quad (3)$$

$$k_b \leq N, \quad (4)$$

Stage 1 (Ensemble of DNNs Trained to Defend Against Adversarial Examples)

Stage 2 (Our Proposed Window-Based Algorithm OSRA)

Fig. 2: An example shows how OSRA works

$$k_{i+1} - k_i \geq w + 1, 1 \leq i \leq b - 1, \quad (5)$$

 Particularly, for $p = 1$, it is reduced to

$$\underbrace{k_1}_{\geq w+p} + \sum_{i=1}^{b-1} \left(\underbrace{k_{i+1} - k_i}_{\geq w+1} \right) \leq N - p + 1 \quad (6)$$

$$N \geq b(w + 1), \quad (8)$$

Thus,

$$N - p + 1 \geq (b - 1)(w + 1) + w + p. \quad (7)$$

- Let $\mathcal{P}_{w,b}^{(1,N)}$ be the probability to have the event $\mathcal{E}_{w,b}^{(1,N)}$ occurred.

$$\mathcal{P}_{w,b}^{(1,N)} = \Pr \left(\mathcal{E}_{w,b}^{(1,N)} \right). \quad (9)$$

B. Statistical properties of X_i and Z_k

- Probability density function (PDF)

$$f_{X_i}(x) = \begin{cases} 1, & x \in [0, 1] \\ 0, & \text{elsewhere} \end{cases}, \quad (10)$$

- Cumulative distribution function (CDF)

$$F_{X_i}(x) = \begin{cases} x, & x \in [0, 1] \\ 0, & \text{elsewhere} \end{cases}, \quad (11)$$

- CDF of $\min(X_{i_1}, X_{i_2}, \dots, X_{i_k})$ with $i_j \in \{1, \dots, N\}$

$$F_{\min(X_{i_1}, X_{i_2}, \dots, X_{i_k})}(x) = 1 - \prod_{j=1}^k (1 - F_{X_{i_j}}(x)) \quad (12)$$

$$= \begin{cases} 1 - (1 - x)^k, & x \in [0, 1] \\ 0, & \text{elsewhere} \end{cases}.$$

C. Evaluation of $\mathcal{P}_{w,b}^{(1,N)}$

In what follows, a closed-form of the probability defined in (9) is provided. As such, the optimum value w^* of the window size for a given value of the budget b and the number of elements N is determined, i.e.,

$$w^* = \arg \max_{w=1.. \lfloor \frac{N}{b} \rfloor - 1} \mathcal{P}_{w,b}^{(1,N)}. \quad (13)$$

It may be deduced that such a probability is necessary to compute w^* to this purpose. As a result, the following Theorem yields a recursive formula that links $\mathcal{P}_{w,b}$ and $\mathcal{P}_{w,b-1}$. In Theorem 2, which closes the computing process, the same probability corresponding to $b = 1$ is given.

In other words, Theorem 2 is a base theorem that contributes to calculating the optimal window size for N elements when the allocated budget for crowdsourcing $b = 1$, whereas Theorem 1 is a generalization for theorem 2. It contributes to calculating the optimal window size for N elements when the allocated budget for crowdsourcing is more than 1 (i.e., $b > 1$).

Theorem 1. For an arbitrary positive number b , the following induction formula holds

$$\mathcal{P}_{w,b}^{(1,N)} = \begin{cases} 0 & , N < b(w+1) \\ \frac{1}{(w+1)^b} & , N = b(w+1) \\ \mathcal{P}_{w,1}^{(1,N)} & , b = 1 \\ \frac{\mathcal{P}_{w,b-1}^{(N-w-1)}}{w+1} + \sum_{k=2}^{N-(b-1)(w+1)-w} \mathcal{P}_{w,b-1}^{(N-k-w)} \\ \times (\mathcal{B}_{w+1,w+k-1} - \mathcal{B}_{w+1,w+k}), & b > 1 \& \frac{N}{b} > w+1 \end{cases} \quad (14)$$

where

$$\mathcal{B}_{i,j} = \begin{cases} \frac{w}{w+1}, & i = j \\ \frac{w}{w+1} - \sum_{\ell=i}^{j-1} \mathcal{C}_{\ell,j}, & j > i \end{cases}, \quad (15)$$

and $\mathcal{C}_{\ell,j}$ can be computed by induction as

$$\mathcal{C}_{\ell,j} = \begin{cases} \frac{1}{(j-\ell+2) \times (j-\ell)!}, & w = 1 \\ \frac{1}{j-\ell+w+1}, & \ell + w \geq j \\ \frac{w}{2(w+1)^2}, & \ell + w + 1 = j \\ \frac{\frac{w}{w+1} - \sum_{q=\ell+w+1}^{j-1} \mathcal{C}_{q,j}}{j-\ell+w+1}, & \ell + w < j - 1 \end{cases}, w \geq 2 \quad (16)$$

Proof: First, it is worthwhile to mention that $\mathcal{E}_{w,b}^{(1,N)}$ is equivalent to the following event

$$\mathcal{E}_{w,b}^{(1,N)} = \left\{ \mathcal{E}_{w,1}^{(N_1-w, N_1)} \cap \mathcal{E}_{w,b-1}^{(N_1+1, N)} \right\}_{w+1 \leq N_1 \leq N - (b-1)(w+1)}. \quad (17)$$

- On the other hand, if $N = b(w+1)$, then the following holds

$$\mathcal{P}_{w,b}^{(1,b(w+1))} = \frac{1}{(w+1)^b}, \quad (18)$$

- If $N > b(w+1)$ then such a probability can be evaluated leveraging (17) as

$$\begin{aligned} \mathcal{P}_{w,1}^{(1,N)} &= \Pr(\overline{\mathcal{A}_{w+1}}, \mathcal{E}_{w,b-1}^{(w+2, N)}) \\ &+ \sum_{k=2}^{N-(b-1)(w+1)-w} \Pr(\mathcal{F}_w^{(k)}, \mathcal{E}_{w,b-1}^{(k+w+1, N)}) \\ &= \frac{\mathcal{P}_{w,b-1}^{(w+2, N)}}{w+1} + \sum_{k=2}^{N-(b-1)(w+1)-w} \mathcal{P}_{w,b-1}^{(k+w+1, N)} \\ &\times \underbrace{\Pr(\mathcal{F}_w^{(k)}, \overline{\mathcal{A}_{w+k}})}_{=\mathcal{B}_{w+1,w+k-1} - \mathcal{B}_{w+1,w+k}}, \end{aligned} \quad (19)$$

where

$$\mathcal{F}_w^{(k)} = \mathcal{A}_{w+1}, \dots, \mathcal{A}_{w+k-1}, \overline{\mathcal{A}_{w+k}}, \quad (20)$$

and

$$\mathcal{B}_{i,j} = \Pr(\mathcal{A}_i, \dots, \mathcal{A}_j), j \geq i. \quad (21)$$

As the RVs in the ranges $[1, w+k-1]$ and $[w+k, N]$ are i.i.d, and for the sake of notations' simplicity, such intervals are represented by the number of their elements instead. That is, $\mathcal{P}_{w,b}^{(1,N)}$ can be denoted as $\mathcal{P}_{w,b}^{(N)}$ for arbitrary values of w, b , and N . Therefore (14) is attained.

To finalize the Theorem 1's proof, it is sufficient to evaluate $\mathcal{B}_{i,j}$. Towards this end, let's distinguish two cases:

1) *Case 1.* $i = j$: Using the definitions (1) and (2) alongside (10) and (12), one obtains

$$\begin{aligned} \mathcal{B}_{i,i} &= \Pr(X_i > Z_i) \\ &= \int_{x=0}^1 F_{Z_i}(x) f_{X_i}(x) dx \\ &= \int_{x=0}^1 [1 - (1-x)^w] dx \\ &= \frac{w}{w+1}. \end{aligned} \quad (22)$$

2) *Case 2. $j > i$:* The above probability can be rewritten as

$$\mathcal{B}_{i,j} = \underbrace{\Pr(\mathcal{A}_{i+1}, \dots, \mathcal{A}_j)}_{=\mathcal{B}_{i+1,j}} - \Pr(\overline{\mathcal{A}_i}, \mathcal{A}_{i+1}, \dots, \mathcal{A}_j), i < j. \quad (23)$$

By induction, (23) can be expressed as

$$\mathcal{B}_{i,j} = \mathcal{B}_{j,j} - \sum_{\ell=i}^{j-1} \mathcal{C}_{\ell,j}, j > i - 1. \quad (24)$$

Now Substituting (22) into (24), (15) is attained. Besides,

$$\begin{aligned} \mathcal{C}_{\ell,j} &= \Pr(\mathcal{A}_j, \dots, \mathcal{A}_{\ell+1}, \overline{\mathcal{A}_\ell}) \\ &= \Pr(\mathcal{A}_j, \dots, \mathcal{A}_{\ell+1} | \overline{\mathcal{A}_\ell}) \Pr(\overline{\mathcal{A}_\ell}). \end{aligned} \quad (25)$$

Given that $\overline{\mathcal{A}_\ell}$ is occurring, i.e., $X_\ell < \min(X_{\ell-1}, \dots, X_{\ell-w})$, the event $\mathcal{A}_j, \dots, \mathcal{A}_{\ell+1} | \overline{\mathcal{A}_\ell}$ has two different forms depending on the value of w :

Subcase 1: $w = 1$

Here, $\mathcal{A}_k = \{X_k < X_{k-1}\}, \ell + 1 \leq k \leq j$. It follows that

$$\mathcal{C}_{\ell,j} = \Pr\left(\underbrace{X_\ell < X_{\ell-1}, X_j > X_{j-1} > \dots > X_\ell}_{\mathcal{G}_{\ell,j}}\right). \quad (26)$$

The joint event in (26) can be written as the union of the following $j - \ell + 1$ independent single events

$$\begin{aligned} \mathcal{G}_{\ell,j} &= \bigcup_{k=\ell+1}^j \{X_\ell < \dots < X_{k-1} < X_{\ell-1} < X_k < \dots < X_j\} \\ &\quad \bigcup \{X_\ell < X_{\ell+1} < \dots < X_j < X_{\ell-1}\}. \end{aligned} \quad (27)$$

Therefore, (26) can be written as a summation of the above events' probabilities. Besides, as X_i are i.i.d RVs, these probabilities are equal. Thus, $\mathcal{C}_{\ell,j}$ can be evaluated as

$$\begin{aligned} \mathcal{C}_{\ell,j} &= \sum_{\ell=i}^{j-1} \Pr(X_\ell < \dots < X_{k-1} < X_{\ell-1} < X_k < \dots < X_j) \\ &\quad + \Pr(X_\ell < X_{\ell+1} < \dots < X_j < X_{\ell-1}) \\ &= (j - \ell + 1) \times \int_{x_{\ell-1}=0}^1 \int_{x_j=0}^{x_{\ell-1}} \\ &\quad \dots \int_{x_{\ell+1}=0}^{x_{\ell+2}} \int_{x_\ell=0}^{x_{\ell+1}} \prod_{k=\ell-1}^j \left\{ \underbrace{f_{X_k}(x_k)}_{=1} dx_k \right\} \\ &= (j - \ell + 1) \times \frac{1}{(j - \ell + 1)!} \int_0^1 x_{\ell-1}^{j-\ell+1} dx_{\ell-1} \\ &= \frac{j - \ell + 1}{(j - \ell + 2)!}. \end{aligned} \quad (28)$$

Subcase 2: $w \geq 2$

Under this subcase, and by assuming that $X_\ell < \min(X_{\ell-1}, \dots, X_{\ell-w})$, Z_k defined in (22) is reduced for an index in the range $\ell + 1 \leq k \leq \ell + w$ to

$$\begin{aligned} Z_k &= \min(X_{k-1}, X_{k-2}, \dots, X_{k-w}) \\ &= X_\ell. \end{aligned} \quad (29)$$

It follows that

$$\begin{aligned} \mathcal{C}_{\ell,j} &= \Pr\left(\bigcap_{k=\ell+w+1}^j \{X_j > Z_j\}, X_\ell < \min_{\substack{1-w \leq k \leq w \\ k \neq \ell}} (X_k), \overline{\mathcal{A}_\ell}\right) \\ &= \Pr\left(\bigcap_{k=\ell+w+1}^j \{X_j > Z_j\}, X_\ell < \min_{\substack{\ell-w \leq k \leq \ell+w \\ k \neq \ell}} (X_k)\right) \end{aligned} \quad (30)$$

Consequently, two different subcases can be distinguished depending on the existence of the first event in (30) (i.e., the intersection of events).

• **Subcase 2.1:** $\ell + w \geq j$

In this case, such an intersection of events is not defined. As a result,

$$\begin{aligned} \mathcal{C}_{\ell,j} &= \Pr\left(X_\ell < \underbrace{\min_{\substack{\ell-w \leq k \leq j \\ k \neq \ell}} (X_k)}_{U_{\ell,w}}\right) \\ &= \int_{z=0}^1 (1 - F_{U_{\ell,w}}(x)) f_{X_\ell}(x) dx \\ &\stackrel{(a)}{=} \int_{x=0}^1 (1 - x)^{j-\ell+w} dx \\ &= \frac{1}{j - \ell + w + 1}. \end{aligned} \quad (31)$$

where step (a) holds using (10) and (12).

• **Subcase 2.2:** $\ell + w < j$

In this case, the intersection of events is not empty. That is,

$$\begin{aligned} \mathcal{C}_{\ell,j} &= \Pr\left(X_\ell < \min_{\substack{\ell-w \leq k \leq \ell+w \\ k \neq \ell}} (X_k) | \mathcal{A}_{\ell+w+1}, \dots, \mathcal{A}_j\right) \\ &\quad \times \underbrace{\Pr(\mathcal{A}_{\ell+w+1}, \dots, \mathcal{A}_j)}_{=\mathcal{B}_{\ell+w+1,j}} \\ &= \Pr\left(X_\ell < \min_{\substack{\ell-w \leq k \leq j \\ k \neq \ell}} (X_k)\right) \mathcal{B}_{\ell+w+1,j} \\ &= (j - \ell + w) \mathcal{B}_{\ell+w+1,j} \int_{z=0}^1 z (1 - z)^{j-\ell+w-1} dz \\ &= \frac{\mathcal{B}_{\ell+w+1,j}}{j - \ell + w + 1}. \end{aligned} \quad (32)$$

Now using (31), and (22) alongside (24), (16) is attained, which concludes the proof of Theorem 1. ■

Owing to the above, it is sufficient to evaluate $\mathcal{P}_{w,1}^{(N)}$ with $N > w + 1$ so that to evaluate $\mathcal{P}_{w,m}^{(N)}$ for $m > 1$ according to (14).

Theorem 2. Suppose $b = 1$. The following equation holds

$$\begin{aligned} \mathcal{P}_{w,1}^{(N)} &= \frac{\mathcal{B}_{2w+2,N}}{w+1} + \sum_{k=w+2}^{N-w-1} \mathcal{B}_{k+w+1,N} (\mathcal{B}_{w+1,k-1} - \mathcal{B}_{w+1,k}) \\ &\quad + \mathcal{B}_{w+1,N-w-1} - \mathcal{B}_{w+1,N}. \end{aligned} \quad (33)$$

Proof: First, it can be noticed that $\mathcal{E}_{w,1}^{(N)}$ is equivalent to either $\{\overline{\mathcal{A}_k}, (\mathcal{A}_j)_{j \in \{w+1, \dots, k-1, k+w+1, \dots, N\}}\}_{w+1 \leq k \leq N-w-1}$ or $\{\overline{\mathcal{A}_k}, (\mathcal{A}_j)_{w+1 \leq j \leq k-1, j \neq k}\}_{N-w \leq k \leq N}$. The latter events are corresponding to the last subinterval $[N-w, N]$ of width lesser than $w+1$. Therein, if $\overline{\mathcal{A}_k}$ holds then all events $(\mathcal{A}_j)_{j \geq k+w+1}$ are not defined within the range $[1, N]$, yields

$$\begin{aligned} \mathcal{P}_{w,1}^{(N)} &= \Pr(\overline{\mathcal{A}_{w+1}}, \mathcal{A}_{2w+2}, \dots, \mathcal{A}_N) \\ &+ \sum_{N-w}^N \Pr(\mathcal{A}_{w+1}, \dots, \mathcal{A}_{k-1}, \overline{\mathcal{A}_k}) \\ &+ \sum_{k=w+2}^{N-w-1} \Pr(\mathcal{A}_{w+1}, \dots, \mathcal{A}_{k-1}, \overline{\mathcal{A}_k}, \mathcal{A}_{k+w+1}, \dots, \mathcal{A}_N). \end{aligned} \quad (34)$$

It is worthy to mention that the third term in (34) equals 0 if $N-w-1 < w+2$, i.e., $N < 2w+3$.

Now, by noticing that the set of events $\{\mathcal{A}_{k+w+1}, \dots, \mathcal{A}_N\}$ is independent of $\{\mathcal{A}_{w+1}, \dots, \mathcal{A}_{k-1}, \overline{\mathcal{A}_k}\}$, applying the Bayes Rule, and performing some algebraic operations, (34) becomes

$$\begin{aligned} \mathcal{P}_{w,1}^{(N)} &= \Pr(\overline{\mathcal{A}_{w+1}}) \mathcal{B}_{2w+2,N} + \sum_{N-w}^N \underbrace{\Pr(\mathcal{A}_{w+1}, \dots, \mathcal{A}_{k-1}, \overline{\mathcal{A}_k})}_{=\mathcal{B}_{w+1,k-1}-\mathcal{B}_{w+1,k}} \\ &+ \sum_{k=w+2}^{N-w-1} \Pr(\mathcal{A}_{w+1}, \dots, \mathcal{A}_{k-1}, \overline{\mathcal{A}_k}) \mathcal{B}_{k+w+1,N} \\ &= (1 - \mathcal{B}_{w+1,w+1}) \mathcal{B}_{2w+2,N} + \mathcal{B}_{w+1,N-w-1} - \mathcal{B}_{w+1,N} \\ &+ \sum_{k=w+2}^{N-w-1} (\mathcal{B}_{w+1,k-1} - \mathcal{B}_{w+1,k}) \mathcal{B}_{k+w+1,N}. \end{aligned} \quad (35)$$

■

V. EXPERIMENTS AND RESULTS

A. Performance Evaluation Metric

Given an allocated budget b and assuming that crowdsourcing one element (i.e., validating and/or relabeling an image) costs $1/b$. Let c denote the cost for validation without relabeling (i.e., if a crowdsourcing worker finds that the assigned label is correct), and d denote the cost for validating and relabeling (correcting the initially assigned label). If OSRA manages to validate and relabel c/b , while manages to validate without relabeling d/b , we say that the crowdsourcing process is represented by $(CP) = c/b + d/b$. To measure the performance of OSRA, we consider the budget used in validating and relabeling (i.e., c/b) because it is the only budget that contributes directly to improving the system's accuracy. So, the performance metric is the success rate of relabeling SR (budget utilization). That is, the number of corrected (relabelled) elements out of the overall crowdsourced elements. In this article, we interchangeably use the terms "OSRA's budget utilization" and "OSRA's success rate of relabeling".

Though c contributes to augmenting the reliability of the system in an indirect way (the double-checking makes us more

confident about the validity of the label), we do not consider it as it has no direct effect on our performance metric (success rate of relabeling). Success rate of relabeling SR contributes directly to the overall system's accuracy. System robustness is a combination of state-of-the-art model accuracy (Stage 1) and added robustness (Stage 2). Added Robustness $AR = SR * b$, which means a positive correlation between AR and SR .

B. Experimental Setup

Our work aims at enhancing the robustness of the cutting-edge DNNs against black-box transfer-based attacks. It can complement any relevant state-of-the-art adversarial defense approach. For that, as an example, we built our experiments on top of an ensemble-based adversarial defense technique [16]. In particular, we used a pre-trained ensemble model trained on CIFAR-10 dataset in a novel ensemble training process that maximizes the diversity between each pair of sub-models and minimizes the transferability of adversarial examples. The selected ensemble model comprises of three sub-models each is based on ResNet-20 [45]. The final prediction of the ensemble is the average of the output probabilities of the sub-models.

We applied the ensemble model to a test dataset composed of 1000 adversarial examples corresponding to the original sample of CIFAR-10. The adversarial examples are generated from a surrogate ensemble which was configured with: 3 sub-models, SGM [46] as the attack methodology, CW [18] as the loss function, and attack-strength (epsilon) = 0.01. The stream of the results generated by the ensemble model in the aforementioned experimental setting is used as a base for performing our experiments. We assume that the crowdsourced workers utilized in Stage 2 are reliable (honest and highly skilled).

In all our conducted experiments, to be more confident about the gained results, we repeated each experiment 10^4 times and averaged the results. We also considered the default setting to be a dataset of size 1000 and allocated budget of 10% (i.e., the budget that is enough for crowdsourcing 10

C. Potential Applications for OSRA

Our proposed work is generic and can be applied in many application domains. The work in this article is considered just an example of one use case, i.e., applying OSRA in the robust machine learning domain. OSRA can be applied in other domains. In particular, the mission-critical domains that are not time-sensitive, such as face recognition systems, and x-ray inspection. OSRA can also be used as a training technique in online learning, and as human-intervention-agnostic labeling technique.

X-Ray Inspection: Utilizing radiologists to validate suspected x-ray images is costly. Thus, they should be utilized efficiently. That is, they are given the x-ray images that are most likely having a wrong class. OSRA helps in optimizing such situations. It maximizes the potential of selecting the x-ray images with highly likely wrong prediction class and minimizes the cost of the radiologist as he/she will only get assigned the x-rays that ML model is at most confused about.

Face Recognition: Likewise, OSRA can be applied in face recognition systems. Prediction scores below a certain threshold are considered suspicious, so they are fed into OSRA to select the most suspicious ones to get verified manually by a guardian (human). By this, the allocated budget for human intervention (number of guardians) is utilized wisely, and the robustness of the face recognition system is improved.

Online Learning: ML is going real-time (online learning). The direction of training ML models is going from the traditional approach (batch-based) to the online-based approach. In online learning, the model keeps training on the fly to compensate the potential of the fast change in the data distribution (adapting the model to constantly changing data) and hence ensures a lower error rate of prediction. Our proposed algorithm can involve and help in enhancing this online-based training process by opportunistically selecting the elements with the least prediction confidence and include these elements in the online training process (after validating them through the crowdsourcing process). Reviewing, validating, and correcting the selected elements (i.e. the crowdsourcing process) converts them into a high-quality training data which contributes to enhancing model accuracy. By opportunistically selecting those least confidently predicted elements, the online learning process becomes more efficient. It utilizes the crowdsourcing budget wisely and ensures maintaining the model's higher accuracy with fewer training elements.

OSRA as an Online Active Learning Technique: OSRA can be seen as a corresponding online training technique to offline active learning. They both serve as an efficient selection approach in a budget-constraint situation. In other words, while an active learning technique is used to select and manually label the most uncertain samples in a batch of samples, OSRA can be used to select and manually label the most uncertain samples in a stream of samples (they come as a sequence of one-by-one samples).

Human-Intervention-Agnostic Labeling Technique: OSRA can be utilized by different types of human-intervention labelers such as crowdsourced workers and Subject Matter Experts (SME). Crowdsourced worker is a kind of human intervention where a labeler has an average knowledge of the task he labels, which could be any popular task such as image classification or face recognition. On the other hand, SME is a human intervention labeler where the task needs specialization and deep knowledge such as X-Ray inspection.

OSRA can also be utilized in several other areas, such as enhancing the defenses of cyber-physical systems [47], bearing fault diagnosis [48], and reducing real-world adversarial threats [49].

D. Experimental Results

In this section, we present the results of different experiments conducted in this work. We dedicated Section V-E to go beyond the presented results and show intuitions and interpretations.

1) *OSRA Performance:* We compare OSRA against two baselines including an offline and a random selection algorithm. The offline selection algorithm achieves the best

performance because all inputs are assumed to be available before the selection process. Hence, it needs only sorting the elements (e.g., ascending) and selects the first portion of the elements based on the available budget. As shown in Fig. 3, our proposed algorithm achieves higher performance (success rate of relabeling) than that of the random selection algorithm. It also achieves competitive performance to the offline algorithm. The more success rate of relabeling we have, the more adversarial examples are captured and fixed (corrected labels), and thus, the more added robustness we can achieve. That is, added robustness and budget utilization are linked with each other (further details in Section V-D5).

Fig. 3: Comparing the success rate of relabeling of our proposed algorithm (OSRA) against offline and random selection algorithms. It can be seen that OSRA achieves higher performance than that the random selection algorithm and comparable performance to the offline selection algorithm (the offline benchmark).

2) *OSRA's Optimal Sliding Window Size:* To empirically investigate the optimality of the window-size suggested by our proposed algorithm (the theoretical proof is mentioned in Section IV), we performed a set of experiments using various window sizes: 1) two window sizes lower than the suggested window size w (lower by 20% and 40%, respectively). 2) Two window-sizes larger than the suggested window-size (higher by 20% and 40%, respectively). We observed that OSRA performance SR is higher when we conducted the experiments using the window of size w (the OSRA suggested window size) as illustrated in Fig. V-D2. At the core of OSRA is suggesting the optimal sliding window size, hence, the factors that have impact on window size are necessarily affecting OSRA.

3) *The Impact of Data size and Allocated Budget on OSRA:* The suggested sliding window size depends on N and b (i.e., the stream of output confidences coming from the Stage 1 and the allocated budget). To inspect the impact of each of which on the OSRA performance, we investigated various dataset sizes: 1000, 800, 600, 400, 200 (fixing allocated budget to 10% and window size to w), and different allocated budgets b : 3%, 6%, 9%, 12%, 15% (fixing dataset size to 1000 and window size to w).

We observe that the change in data size N has a trivial impact on the success rate of relabeling SR , i.e., there is no correlation between N and SR . On the other hand, there is a strong negative correlation between allocated budget b and SR . That is, the more allocated budget is, the less success

Fig. 4: Investigating the optimality of the suggested sliding window by checking various window sizes. We can see that the window size suggested by OSRA is optimal. It efficiently leverages the allocated budget to crowdsource and validate the most suspicious elements (the potential adversarial examples).

rate of relabeling is (see Fig. 5). One interpretation is that our proposed algorithm is optimized for reducing the allocated budget (it encourages using smaller budget). Further related discussion is in Section V-E.

Fig. 5: The impact of allocated budget on success rate of relabeling (considering our algorithm’s suggested window and dataset size 1000). We observe a strong negative correlation between the allocated budget and the success rate of relabeling. The more allocated budget is, the less success rate of relabeling is. OSRA is optimized to encourage using less budget.

For more convenience and to easily observe the impact of the allocated budget b , the sliding window-size w , and the size of the dataset on the success rate of relabeling, we illustrated them together in Fig. 6. We have conducted the experiments on various allocated crowdsourcing budgets (5%, 10%, 15%), and various sliding window-sizes (bigger and smaller than w by 20% and 40%, respectively) on datasets of size 1000, 800, and 600. We found that the more allocated budget b , the less performance achieved (less success rate of relabeling). We also found that the larger sliding window size is, the higher performance is. That is, there is a negative correlation between the allocated budget and success rate of relabeling (b and SR), and a positive correlation between sliding window size and success rate of relabeling (W and SR). A similar trend was shown when explored with different dataset sizes. That means that no strong correlation between the dataset size and OSRA performance.

4) *Enhancing OSRA with a Dataset Heuristic Threshold Selection Method:* To enhance the performance of OSRA,

(a) Investigating the success rate of relabeling for various allocated budgets and window sizes when dataset size is 1000.

(b) Investigating the success rate of relabeling for various allocated budgets and window sizes when dataset size is 800.

(c) Investigating the success rate of relabeling for various allocated budgets and window sizes when dataset size is 600.

Fig. 6: Investigating the success rate of relabeling for various allocated budgets and window sizes on different dataset sizes.

we proposed heuristic threshold selection to filter the most suspicious elements in the dataset. We conducted a set of experiments that show the difference in success rate of relabeling when we consider the dataset with and without applying the threshold. We repeated the experiments conducted in Fig. 3 but with a threshold. We noticed that the negative correlation between the allocated budget and success rate of relabeling (b and SR) and the positive correlation between sliding window size, and success rate of relabeling (W and SR) became higher when we applied the proposed threshold. As illustrated in Fig. 7, OSRA utilizes almost 70% of the allocated budget comparing with 85% for the benchmark (the offline algorithm). This shows that our proposed heuristic threshold scheme can significantly enhance the performance of OSRA.

5) *The Impact of OSRA’s Performance on System Robustness:* In our work, the system’s robustness is determined by the accuracy of a state-of-the-art model (Stage 1) and the

Fig. 7: Comparing the success rate of relabeling of our proposed algorithm (OSRA) against offline and random selection algorithms using the proposed dataset heuristic threshold selection method. It can be seen that proposed heuristic threshold can significantly enhance OSRA performance. When we apply the proposed heuristic threshold, the negative correlation between the allocated budget and the success rate of relabeling becomes higher, likewise the positive correlation between the window size and the success rate of relabeling.

added robustness (Stage 2). The added robustness (AR) is what our proposed algorithm OSRA contributes to the overall robustness of the system. AR is defined with respect to success rate of relabeling SR as $AR = SR * d$. Given that the $SR = c/d$ (as explained in Section V-A), it can be seen that the added robustness AR has a positive correlation with SR (i.e., the more success rate of relabeling is, the more added robustness is). For example, if SR is 70%, and the b is 10 (i.e., the budget that is enough for crowdsourcing 10% of elements in the stream), AR will be 7%. If the state-of-the-art accuracy is 90%, the overall robustness will be 97% (90% + 7%). Likewise if SR is 80%, the added robustness will 8%, and consequently, the system robustness becomes 98%, and so on. As success rate of relabeling is not an independent variable (it depends on b , N , and w), to investigate its impact on the added robustness, we investigate the dependent factors. Thus, based on the aforementioned correlation between SR and AR , in the set of experiments (Fig. 3, Fig. 4, Fig. 7), where we were investigating one factor each time, the impact of those investigated factors (Sections: V-A, V-D2, V-D4) on AR will contribute directly in a proportional manner to AR . For example, as shown in Fig. 8 when we investigate b considering variant values, we observe a positive correlation with AR . That is, the more allocated budget is, the more added robustness is.

E. Discussion

In this section, we go beyond the reported results in Section V and provide intuitions and interpretations to the reported results. Online selection algorithm cannot meet the performance of an offline selection algorithm. An online algorithm is called a competitive algorithm in case the proportion between the performance of online and offline algorithms is bounded [21]. In Section V-D1, we noticed that the performance of OSRA is competitive to the performance of the offline selection algorithm. By repeating the experiments V-D1 10^4 times, we

Fig. 8: Investigating the impact of allocated budget (as an example of the factors that contribute to success rate of relabeling) on the system robustness. There is a positive correlation between the allocated budget and the added robustness to the system.

found that the ratio between our proposed algorithm (OSRA) and an optimal offline algorithm is 0.67.

The results in Section V-D2 show that success rate of relabeling is higher when we conducted the experiments using the optimal window size w (i.e., the one suggested by our proposed algorithm). Increasing window size by a certain percentage means that the potential of selecting more elements (that fulfill the allocated budget b) is lower (one selection per window). In other words, the allocated budget may not be used completely as the total number of the selected elements might be lower than the available budget. Likewise added robustness is lower when we decrease the window size. A lower sliding window size w means a lower potential of finding a suspicious element. Thus, optimal window size should be large enough to select a suspicious element and small enough to allow for selecting elements that meet the allocated budget b .

The results in Section V-D3 demonstrate a negative correlation between the allocated budget b and the success rate of relabeling SR . The results also demonstrate a positive correlation between w and SR . This occurs because the higher b means the lower size of the suggested window w (that is how our proposed algorithm adapts to select a suspected element from each window) as explained in the Section IV. So, a lower window size means that the potential of finding a valid suspected element is lower and vice versa.

The results in Section V-D3 show that there is a trivial impact of dataset size N on success rate of relabeling SR . This trivial impact resulted from the fact that dataset size proportionally relates to the allocated budget (we consider it as a percentage from N). That is, given an allocated budget 10%, as an example, means allocating a budget that is enough for crowdsourcing 10% of dataset size. Assuming that one budget unit is equivalent for crowdsourcing one dataset element (as explained in V-A), the allocated budget 10% from dataset of size 1000, 500, 100 will be proportionally the same. So, whenever we increase or decrease the dataset size, the allocated budget will change accordingly.

Comparing the results in Section V-D1 with the results in Section V-D4 where a heuristic threshold selection method was applied, we observe a remarkable improvement in the success

rate of relabeling. It narrows down the space of the suspicious elements, which increases the probability of selecting the most suspicious elements. Without filtering the suspected elements, there will be many cases where our proposed algorithm selects less suspected elements (elements that have the lowest output confidence within the sliding window, however, in reality they are with higher output confidences (not suspected). Hence, the elements corresponding to these lower output confidences get selected to be crowdsourced. However, when a crowdsourced worker validates an element and finds that Stage 1 has already predicted it correctly, it means that the allocated budget for crowdsourcing this element is not utilized. That is, it is not involved in improving budget utilization (the number of relabeled elements plays a crucial role in calculating the budget utilization). This kind of validation contributes to making the system more robust in an indirect way, though. Ensuring that elements are predicted successfully is a kind of manual double-checking that increases the trust in the overall system prediction. However, in our work, we only care about the success rate of relabeling (budget utilization), which is the performance metric.

F. Lessons Learned

The key lessons learned from the experiments can be summarized as follows (they can be seen as brief corresponding answers to the research questions raised in Section I).

- Our proposed online algorithm achieved higher performance than that of the random selection algorithm and comparable performance to the offline selection algorithm (the offline benchmark).
- The conducted experiments showed that the window size suggested by our proposed algorithm (OSRA) is optimal.
- The change in dataset size has a trivial impact on the success rate of relabeling, i.e., there is no correlation between dataset size and success rate of relabeling.
- Our proposed heuristic threshold can significantly enhance OSRA performance.
- The more allocated budget, the less window size is suggested by (OSRA), and the less window size, the less budget utilization (less success rate of relabeling). When the window size is less, the probability of finding a suspicious element get reduced.
- The more success rate of relabeling, the more corrected labels and hence, the more enhancement in the system robustness (more added robustness).

VI. CONCLUSIONS AND FUTURE WORK

In this work, we propose an Online Selection and Relabeling Algorithm namely OSRA, which is integrated into a state-of-the-art defense technique resulting in a more robust prediction system. We empirically proved the optimality and efficiency of OSRA in crowdsourcing the most suspicious elements (the potential adversarial examples) the defense technique failed to capture.

The cutting-edge adversarial ML defenses were found not wholly robust. Hence, they need to augment their reliability by another layer of protection. We empirically evaluated

OSRA by comparing its performance with two baselines, an offline and random selection algorithms. We investigated the optimality of the suggested sliding window size, and checked its impact on various dataset sizes and allocated budgets. Then, we enhanced OSRA by proposing a heuristic dataset threshold selection method.

In our future work, we plan to investigate the idea of an adaptive sliding window. By considering OSRA's suggested window size as the default size for the initial comparison process, the subsequent sliding windows will have the adaptive size and the sliding steps based on whether or not suspected elements were found in the proceeding sliding windows.

VII. ACKNOWLEDGMENT

This publication was made possible by NPRP grant # [13S-0206-200273] from the Qatar National Research Fund (a member of Qatar Foundation). The statements made herein are solely the responsibility of the authors.

REFERENCES

- [1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 25, pp. 1097–1105, 2012.
- [2] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7263–7271.
- [3] K. Khan, R. U. Khan, K. Ahmad, F. Ali, and K.-S. Kwak, "Face segmentation: A journey from classical to deep learning paradigm, approaches, trends, and directions," *IEEE Access*, vol. 8, pp. 58 683–58 699, 2020.
- [4] M. Malik, M. K. Malik, K. Mehmood, and I. Makhdoom, "Automatic speech recognition: a survey," *Multimedia Tools and Applications*, vol. 80, no. 6, pp. 9411–9457, 2021.
- [5] D. W. Otter, J. R. Medina, and J. K. Kalita, "A survey of the usages of deep learning for natural language processing," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 2, pp. 604–624, 2020.
- [6] I. Alsmadi, K. Ahmad, M. Nazzal, F. Alam, A. Al-Fuqaha, A. Khreishah, and A. Algosaihi, "Adversarial attacks and defenses for social network text processing applications: Techniques, challenges and future research directions," *arXiv preprint arXiv:2110.13980*, 2021.
- [7] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.
- [8] A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry, "Adversarial examples are not bugs, they are features," *arXiv preprint arXiv:1905.02175*, 2019.
- [9] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.
- [10] D. Meng and H. Chen, "Magnet: a two-pronged defense against adversarial examples," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 135–147.
- [11] Y. Song, T. Kim, S. Nowozin, S. Ermon, and N. Kushman, "Pixeldefend: Leveraging generative models to understand and defend against adversarial examples," *arXiv preprint arXiv:1710.10766*, 2017.
- [12] G. Katz, C. Barrett, D. L. Dill, K. Julian, and M. J. Kochenderfer, "Reluplex: An efficient smt solver for verifying deep neural networks," in *International Conference on Computer Aided Verification*. Springer, 2017, pp. 97–117.
- [13] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *2016 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2016, pp. 582–597.
- [14] R. Huang, B. Xu, D. Schuurmans, and C. Szepesvári, "Learning with a strong adversary," *arXiv preprint arXiv:1511.03034*, 2015.
- [15] M. Lecuyer, V. Atlidakis, R. Geambasu, D. Hsu, and S. Jana, "Certified robustness to adversarial examples with differential privacy," in *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2019, pp. 656–672.

- [16] H. Yang, J. Zhang, H. Dong, N. Inkawhich, A. Gardner, A. Touchet, W. Wilkes, H. Berry, and H. Li, "DVERGE: diversifying vulnerabilities for enhanced robust generation of ensembles," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., 2020. [Online]. Available: <https://tinyurl.com/epe3zs3>
- [17] S. Chen, N. Carlini, and D. Wagner, "Stateful detection of black-box adversarial attacks," in *Proceedings of the 1st ACM Workshop on Security and Privacy on Artificial Intelligence*, 2020, pp. 30–39.
- [18] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *IEEE Symposium on Security and Privacy (SP)*, 2017, pp. 39–57.
- [19] I. Goodfellow and N. Papernot. Is attacking machine learning easier than defending it? [Online]. Available: <http://www.cleverhans.io/security/privacy/ml/2017/02/15/why-attacking-machine-learning-is-easier-than-defending-it.html>
- [20] I. Goodfellow, P. McDaniel, and N. Papernot, "Making machine learning robust against adversarial inputs," *Communications of the ACM*, vol. 61, no. 7, pp. 56–66, 2018.
- [21] R. M. Karp, "On-line algorithms versus off-line algorithms: How much is it worth to know the future?" in *IFIP Congress (1)*, vol. 12, 1992, pp. 416–429.
- [22] A. Kovashka, O. Russakovsky, L. Fei-Fei, and K. Grauman, "Crowdsourcing in computer vision," *arXiv preprint arXiv:1611.02145*, 2016.
- [23] J. Buckman, A. Roy, C. Raffel, and I. Goodfellow, "Thermometer encoding: One hot way to resist adversarial examples," in *International Conference on Learning Representations*, 2018.
- [24] C. Guo, M. Rana, M. Cisse, and L. Van Der Maaten, "Countering adversarial images using input transformations," *arXiv preprint arXiv:1711.00117*, 2017.
- [25] P. Samangouei, M. Kabkab, and R. Chellappa, "Defense-gan: Protecting classifiers against adversarial attacks using generative models," *arXiv preprint arXiv:1805.06605*, 2018.
- [26] A. Ilyas, A. Jalal, E. Asteri, C. Daskalakis, and A. G. Dimakis, "The robust manifold defense: Adversarial training using generative models," *arXiv preprint arXiv:1712.09196*, 2017.
- [27] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.
- [28] T. Pang, K. Xu, C. Du, N. Chen, and J. Zhu, "Improving adversarial robustness via promoting ensemble diversity," in *International Conference on Machine Learning*. PMLR, 2019, pp. 4970–4979.
- [29] S. Kariyappa and M. K. Qureshi, "Improving adversarial robustness of ensembles with diversity training," *arXiv preprint arXiv:1901.09981*, 2019.
- [30] G. S. Dhillon, K. Azizzadenesheli, Z. C. Lipton, J. Bernstein, J. Kossaifi, A. Khanna, and A. Anandkumar, "Stochastic activation pruning for robust adversarial defense," *arXiv preprint arXiv:1803.01442*, 2018.
- [31] C. Xie, J. Wang, Z. Zhang, Z. Ren, and A. Yuille, "Mitigating adversarial effects through randomization," *arXiv preprint arXiv:1711.01991*, 2017.
- [32] A. Athalye, N. Carlini, and D. Wagner, "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples," in *International Conference on Machine Learning*. PMLR, 2018, pp. 274–283.
- [33] F. Tramer, N. Carlini, W. Brendel, and A. Madry, "On adaptive attacks to adversarial example defenses," *arXiv preprint arXiv:2002.08347*, 2020.
- [34] D. Hendrycks and K. Gimpel, "Early methods for detecting adversarial images," *arXiv preprint arXiv:1608.00530*, 2016.
- [35] J. H. Metzen, T. Genewein, V. Fischer, and B. Bischoff, "On detecting adversarial perturbations," *arXiv preprint arXiv:1702.04267*, 2017.
- [36] F. Tramèr, "Detecting adversarial examples is (nearly) as hard as classifying them," *arXiv preprint arXiv:2107.11630*, 2021.
- [37] D. Tsipras, S. Santurkar, L. Engstrom, A. Turner, and A. Madry, "Robustness may be at odds with accuracy," *arXiv preprint arXiv:1805.12152*, 2018.
- [38] C. Sitawarin, S. Chakraborty, and D. Wagner, "Sat: Improving adversarial training via curriculum-based loss smoothing," in *Proceedings of the 14th ACM Workshop on Artificial Intelligence and Security*, 2021, pp. 25–36.
- [39] O. Sagi and L. Rokach, "Ensemble learning: A survey," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 8, no. 4, p. e1249, 2018.
- [40] X. Huang, M. Kwiatkowska, S. Wang, and M. Wu, "Safety verification of deep neural networks," in *International Conference on Computer Aided Verification*. Springer, 2017, pp. 3–29.
- [41] N. Carlini and D. Wagner, "Adversarial examples are not easily detected: Bypassing ten detection methods," in *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, 2017, pp. 3–14.
- [42] M. Juuti, S. Szyller, S. Marchal, and N. Asokan, "Prada: protecting against dnn model stealing attacks," in *2019 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2019, pp. 512–527.
- [43] J. M. Cohen, E. Rosenfeld, and J. Z. Kolter, "Certified adversarial robustness via randomized smoothing," *arXiv preprint arXiv:1902.02918*, 2019.
- [44] G. Yang, T. Duan, E. Hu, H. Salman, I. Razenshteyn, and J. Li, "Randomized smoothing of all shapes and sizes," *arXiv preprint arXiv:2002.08118*, 2020.
- [45] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.
- [46] D. Wu, Y. Wang, S.-T. Xia, J. Bailey, and X. Ma, "Skip connections matter: On the transferability of adversarial examples generated with resnets," *arXiv preprint arXiv:2002.05990*, 2020.
- [47] Y. Pang, H. Xia, and M. J. Grimble, "Resilient nonlinear control for attacked cyber-physical systems," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 6, pp. 2129–2138, 2020.
- [48] Z.-H. Liu, B.-L. Lu, H.-L. Wei, L. Chen, X.-H. Li, and M. Rätzsch, "Deep adversarial domain adaptation model for bearing fault diagnosis," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 7, pp. 4217–4226, 2021.
- [49] W. Luo, J. L, K. Liu, and L. Chen, "Learning-based policy optimization for adversarial missile-target assignment," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–12, 2021.